

Texts in Statistical Science

Stochastic Processes with R

An Introduction



Olga Korosteleva



CRC Press

Taylor & Francis Group

A CHAPMAN & HALL BOOK

Stochastic Processes with R

CHAPMAN & HALL/CRC

Texts in Statistical Science Series

Joseph K. Blitzstein, *Harvard University, USA*

Julian J. Faraway, *University of Bath, UK*

Martin Tanner, *Northwestern University, USA*

Jim Zidek, *University of British Columbia, Canada*

Recently Published Titles

Bayesian Thinking in Biostatistics

Gary L. Rosner, Purushottam W. Laud, and Wesley O. Johnson

Linear Models with Python

Julian J. Faraway

Modern Data Science with R, Second Edition

Benjamin S. Baumer, Daniel T. Kaplan, and Nicholas J. Horton

Probability and Statistical Inference

From Basic Principles to Advanced Models

Miltiadis Mavrakakis and Jeremy Penzer

Bayesian Networks

With Examples in R, Second Edition

Marco Scutari and Jean-Baptiste Denis

Time Series

Modeling, Computation, and Inference, Second Edition

Raquel Prado, Marco A. R. Ferreira and Mike West

Foundations of Statistics for Data Scientists

With R and Python

Alan Agresti and Maria Kateri

Fundamentals of Causal Inference

With R

Babette A. Brumback

Stochastic Processes with R

An Introduction

Olga Korosteleva

For more information about this series, please visit: <https://www.routledge.com/Chapman--HallCRC-Texts-in-Statistical-Science/book-series/CHTEXSTASCI>

Stochastic Processes with R

An Introduction

Olga Korosteleva



CRC Press

Taylor & Francis Group

Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

A CHAPMAN & HALL BOOK

First edition published 2022
by CRC Press
6000 Broken Sound Parkway NW, Suite 300, Boca Raton, FL 33487-2742

and by CRC Press
2 Park Square, Milton Park, Abingdon, Oxon, OX14 4RN

CRC Press is an imprint of Taylor & Francis Group, LLC

© 2022 Olga Korosteleva

Reasonable efforts have been made to publish reliable data and information, but the author and publisher cannot assume responsibility for the validity of all materials or the consequences of their use. The authors and publishers have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission to publish in this form has not been obtained. If any copyright material has not been acknowledged please write and let us know so we may rectify in any future reprint.

Except as permitted under U.S. Copyright Law, no part of this book may be reprinted, reproduced, transmitted, or utilized in any form by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying, microfilming, and recording, or in any information storage or retrieval system, without written permission from the publishers.

For permission to photocopy or use material electronically from this work, access www.copyright.com or contact the Copyright Clearance Center, Inc. (CCC), 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400. For works that are not available on CCC please contact mpkbookspermissions@tandf.co.uk

Trademark notice: Product or corporate names may be trademarks or registered trademarks and are used only for identification and explanation without intent to infringe.

ISBN: 978-1-032-15373-5 (hbk)
ISBN: 978-1-032-15473-2 (pbk)
ISBN: 978-1-003-24428-8 (ebk)

DOI: [10.1201/9781003244288](https://doi.org/10.1201/9781003244288)

Publisher's note: This book has been prepared from camera-ready copy provided by the author.

Access the Solutions Manual for instructors: <https://www.routledge.com/9781032153735>

Contents

Preface	vii
Author	ix
1 Stochastic Process, Discrete-time Markov Chain	1
1.1 Definition of Stochastic Process	1
1.2 Discrete-time Markov Chain	1
1.3 Chapman-Kolmogorov Equations	3
1.4 Classification of States	5
1.5 Limiting Probabilities	7
1.6 Computations in R	9
1.7 Simulations in R	14
1.8 Applications of Markov Chain	28
Exercises	37
2 Random Walk	43
2.1 Definition of Random Walk	43
2.2 Must-Know Facts About Random Walk	45
2.3 Simulations in R	47
2.4 Applications of Random Walk	51
Exercises	57
3 Poisson Process	61
3.1 Definition and Must-Know Facts About Poisson Process . .	61
3.2 Simulations in R	66
3.3 Applications of Poisson Process	70
Exercises	76
4 Nonhomogeneous Poisson Process	81
4.1 Definition of Nonhomogeneous Poisson Process	81
4.2 Simulations in R	84
4.3 Applications of Nonhomogeneous Poisson Process	91
Exercises	102

5	Compound Poisson Process	105
5.1	Definition of Compound Poisson Process	105
5.2	Simulations in R	107
5.3	Applications of Compound Poisson Process	111
	Exercises	113
6	Conditional Poisson Process	117
6.1	Definition of Conditional Poisson Process	117
6.2	Simulations in R	119
6.3	Applications of Conditional Poisson Process	123
	Exercises	125
7	Birth-and-Death Process	129
7.1	Definition of Birth-and-Death Process	129
7.2	Simulations in R	132
7.3	Applications of Birth-and-Death Process	135
	Exercises	137
8	Branching Process	141
8.1	Definition of Branching Process	141
8.2	Simulations in R	145
8.3	Applications of Branching Process	148
	Exercises	149
9	Brownian Motion	153
9.1	Definition of Brownian Motion	153
9.2	Processes Derived from Brownian Motion	156
	9.2.1 Brownian Bridge	156
	9.2.2 Brownian Motion with Drift and Volatility	157
	9.2.3 Geometric Brownian Motion	157
	9.2.4 The Ornstein-Uhlenbeck Process	158
9.3	Simulations in R	159
9.4	Applications of Brownian Motion	167
	Exercises	177
	Recommended Books	183
	List of Notations	185
	Index	187

Preface

This book was written as a textbook for an undergraduate, senior-level course on random processes for Statistics majors. The presentation is meant to be light yet sufficiently mathematical, with good, interesting, scientific applications. More advanced topics, such as sigma algebras, martingales, general renewal processes, Levy processes, and stochastic calculus, are deliberately avoided. The knowledge of statistics is limited to a linear regression, goodness of fit test, and point estimation.

There are nine chapters in this book, covering Markov chain, random walk, Poisson processes (homogeneous, nonhomogeneous, compound, and conditional), birth-and-death process, branching process, and Brownian motion. Each chapter gives just enough theory in the form of definitions, propositions, remarks, and examples, followed by a section on simulation of trajectories. Finally, applications of processes are presented. At the end of each chapter, a collection of exercises is included. Some of the exercises are theoretical whereas some others require calculation and simulation.

The R software is used throughout. Complete codes and relevant outputs are given in the text. The website that accompanies this book

<https://home.csulb.edu/~okoroste/stochprocesses.html>

contains complete R codes for all examples and applications, and the data sets in .csv format that are used in applications and/or required for certain exercises. A complete solutions manual is also available to instructors upon request at <https://www.routledge.com/9781032153735>.

Last but not least, I would like to thank John Peach who helped me to simulate a trajectory of a branching process by means of self-referencing functions.

Respectfully,
The author



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Author

Olga Korosteleva, PhD, is a professor of statistics in the Department of Mathematics and Statistics at California State University, Long Beach (CSULB). She earned her Bachelor's degree in mathematics in 1996 from Wayne State University in Detroit, and her PhD in statistics from Purdue University in West Lafayette, Indiana, in 2002. Since then she has been teaching statistics and mathematics courses at CSULB.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Stochastic Process, Discrete-time Markov Chain

1.1 Definition of Stochastic Process

A *stochastic process* $\{X(t), t \in T\}$ is a collection of random variables indexed by a parameter t that belongs to a set T . The parameter t is often referred to as *time*, and the value $X(t)$ is the *state* of the process at time t . The set T is called the *index set* of the process.

The *state space* S of a stochastic process is the set of all possible values of $X(t)$, for any $t \in T$.

If T is a countable set, the stochastic process is termed a *discrete-time process* (or, simply, a *discrete process*). Otherwise, it is called a *continuous-time process*.

1.2 Discrete-time Markov Chain

A *discrete-time Markov chain*¹ is a discrete-time stochastic process $\{X_n, n = 0, 1, 2, \text{etc.}\}$ which state space S is finite or countably infinite and such that

$$\mathbb{P}(X_{n+1} = j \mid X_0 = i_0, X_1 = i_1, \dots, X_n = i) = \mathbb{P}(X_{n+1} = j \mid X_n = i) = P_{ij}, \quad (1.1)$$

that is, the conditional probability of the process being in state j at time $n+1$ given all the previous states depends only on the last-known position (state

¹Introduced in Markov, A. A. (1913). “An example of statistical investigation of the text Eugene Onegin concerning the connection of samples in chains.” (In Russian.) *Bulletin of the Imperial Academy of Sciences of St. Petersburg*, 7(3): 153 — 162.

i at time n). This property is called the *Markovian property* (or the *Markov property*). The probability P_{ij} is called the *one-step transition probability* of the Markov chain. Note that it is constant for given states i and j .

In Markov chains, one-step transition probabilities are typically aggregated into a *one-step transition probability matrix*

$$\mathbf{P} = \begin{bmatrix} P_{00} & P_{01} & P_{02} & \dots \\ P_{10} & P_{11} & P_{12} & \dots \\ \dots & \dots & \dots & \dots \\ P_{i0} & P_{i1} & P_{i2} & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}.$$

EXAMPLE 1.1. Consider a Markov chain with the state space $S = \{1, 2, 3\}$ and transition probability matrix

$$\mathbf{P} = \begin{array}{c} \begin{matrix} & 1 & 2 & 3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix} \end{array}.$$

(a) Starting in state 1, the Markov chain returns to it in one step with probability 0.7, or transitions to state 2 with probability 0.1, or transitions to state 3 with probability 0.2. From state 2, the chain cannot transition to state 1 in a single step since the probability of this event is 0. It can, however, return to state 2 or move to state 3 with probabilities 0.6 and 0.4, respectively. If the chain is in state 3, it will transition to states 1 or 2 with respective probabilities 0.5 and 0.2 or will loop back to state 3 with a probability of 0.3. Note that since the chain must transition to some state, the probabilities in each row necessarily sum up to 1.

(b) Conditional probabilities can be computed using the Markov property. For example, $\mathbb{P}(X_3 = 1 \mid X_0 = 1, X_1 = 2, X_2 = 3) = \mathbb{P}(X_3 = 1 \mid X_2 = 3) = P_{31} = 0.5$.

(c) Joint probabilities can be computed by conditioning and using the Markov property. For example, we want to compute the probability that a Markov chain starts in state 1 at time 0, then transitions into state 2, and then into state 3. We obtain $\mathbb{P}(X_0 = 1, X_1 = 2, X_2 = 3) = \mathbb{P}(X_2 = 3 \mid X_0 = 1, X_1 = 2) \mathbb{P}(X_0 = 1, X_1 = 2) = \mathbb{P}(X_2 = 3 \mid X_1 = 2) \mathbb{P}(X_1 = 2 \mid X_0 = 1) \mathbb{P}(X_0 = 1) = P_{23} \cdot P_{12} \cdot \mathbb{P}(X_0 = 1) = (0.4)(0.1)(1) = 0.04$. \square

1.3 Chapman-Kolmogorov Equations

Consider a Markov chain with finite or countably infinite state space $S = \{0, 1, 2, \dots\}$. For fixed states i and j , the n -step transition probability P_{ij}^n is the probability that a process that is currently in state i will be in state j after n transitions, that is, for any time $m \geq 0$, $P_{ij}^n = \mathbb{P}(X_{m+n} = j | X_m = i)$.

Denote the n -step transition probability matrix by $\mathbf{P}^{(n)}$. As a special case with $n = 1$, $\mathbf{P}^{(1)} = \mathbf{P}$.

Next, we will show that $\mathbf{P}^{(n)} = \mathbf{P}^n$, which indicates that finding the n -step transition probability matrix is tantamount to multiplying the one-step transition probability matrix by itself n times.

The proof is based on the *Chapman-Kolmogorov equations* which assert that for all positive integers n and m , $\mathbf{P}^{(n+m)} = \mathbf{P}^{(n)} \cdot \mathbf{P}^{(m)}$, or, equivalently, $P_{ij}^{n+m} = \sum_{k=0}^{\infty} P_{ik}^n P_{kj}^m$, for any states i and j . Indeed, by the definition of conditional probability,

$$P_{ij}^{n+m} = \mathbb{P}(X_{n+m} = j | X_0 = i) = \frac{\mathbb{P}(X_{n+m} = j, X_0 = i)}{\mathbb{P}(X_0 = i)}.$$

Next, we fix state k in which trajectory of the Markov chain is located after n transitions and sum up the probabilities with respect to k , obtaining:

$$P_{ij}^{n+m} = \sum_{k=0}^{\infty} \frac{\mathbb{P}(X_{n+m} = j, X_n = k, X_0 = i)}{\mathbb{P}(X_0 = i)},$$

which by the definition of conditional probability is equal to

$$= \sum_{k=0}^{\infty} \frac{\mathbb{P}(X_{n+m} = j | X_n = k, X_0 = i) \mathbb{P}(X_n = k, X_0 = i)}{\mathbb{P}(X_0 = i)},$$

and, applying the definition of conditional probability again, we get

$$= \sum_{k=0}^{\infty} \mathbb{P}(X_{n+m} = j | X_n = k, X_0 = i) \mathbb{P}(X_n = k | X_0 = i).$$

Finally, applying the Markov property, we can omit from the history all but the latest known state that we condition on, and deduce that

$$P_{ij}^{n+m} = \sum_{k=0}^{\infty} \mathbb{P}(X_{n+m} = j | X_n = k) \mathbb{P}(X_n = k | X_0 = i) = \sum_{k=0}^{\infty} P_{kj}^m P_{ik}^n.$$

This completes the proof. Now, as a corollary of the Chapman-Kolmogorov equations, we will show that $\mathbf{P}^{(n)} = \mathbf{P}^n$. Applying the method of mathematical induction, we first check that the statement is true for small values of n : $\mathbf{P}^{(2)} = \mathbf{P}^{(1+1)} = \mathbf{P}^{(1)} \cdot \mathbf{P}^{(1)} = \mathbf{P} \cdot \mathbf{P} = \mathbf{P}^2$, and $\mathbf{P}^{(3)} = \mathbf{P}^{(2)} \cdot \mathbf{P} = \mathbf{P}^3$. Assuming further that the statement holds for $n - 1$, we prove it for n . We write, $\mathbf{P}^{(n)} = \mathbf{P}^{(n-1)} \cdot \mathbf{P} = \mathbf{P}^{n-1} \cdot \mathbf{P} = \mathbf{P}^n$.

EXAMPLE 1.2. In Example 1.1, we considered a Markov chain with the one-step transition probability matrix

$$\mathbf{P} = \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}.$$

(a) Suppose we would like to find a three-step transition probability matrix. We can compute

$$\begin{aligned} \mathbf{P}^{(3)} = \mathbf{P}^3 &= \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix} \cdot \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix} \cdot \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix} \\ &= \begin{bmatrix} 0.59 & 0.17 & 0.24 \\ 0.20 & 0.44 & 0.36 \\ 0.50 & 0.23 & 0.27 \end{bmatrix} \cdot \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix} = \begin{bmatrix} 0.533 & 0.209 & 0.258 \\ 0.320 & 0.356 & 0.324 \\ 0.485 & 0.242 & 0.273 \end{bmatrix}. \end{aligned}$$

(b) Now we can compute probabilities that require the knowledge of the three-step transition probability matrix. For instance, if $\mathbb{P}(X_0 = 1) = 1$, we can calculate $\mathbb{P}(X_0 = 1, X_1 = 2, X_4 = 3) = \mathbb{P}(X_4 = 3 | X_0 = 1, X_1 = 2) \mathbb{P}(X_1 = 2 | X_0 = 1) \mathbb{P}(X_0 = 1) = \mathbb{P}(X_4 = 3 | X_1 = 2) \mathbb{P}(X_1 = 2 | X_0 = 1) \mathbb{P}(X_0 = 1) = P_{23}^{(3)} \cdot P_{12} \cdot \mathbb{P}(X_0 = 1) = (0.324)(0.1)(1) = 0.0324$. \square

Further, suppose we are given the distribution of the initial state $p_i^0 = \mathbb{P}(X_0 = i)$. Conditioning on this distribution, we can obtain the *unconditional* distribution of the state at time n as

$$p_j^n = \mathbb{P}(X_n = j) = \sum_{i=0}^{\infty} \mathbb{P}(X_n = j | X_0 = i) \mathbb{P}(X_0 = i) = \sum_{i=0}^{\infty} P_{ij}^n p_i^0.$$

In the matrix form, this equation becomes

$$(p_1^n, p_2^n, \dots) = (p_1^0, p_2^0, \dots) \mathbf{P}^{(n)}.$$

EXAMPLE 1.3. Suppose that in Example 1.1 the initial states are equiprobable, that is, $p_1 = p_2 = p_3 = 1/3$. Then, after three transitions, the

probability that the chain ends up in state 1 is $\mathbb{P}(X_3 = 1) = \sum_{i=1}^3 P_{i1}^{(3)} p_i = (0.533)(1/3) + (0.320)(1/3) + (0.485)(1/3) = 0.446$. Note that in these calculations, the three-step transition probabilities come from the first column of the three-step transition probability matrix obtained in Example 1.2. Likewise, the probability that after three steps the chain will be in state 2 can be found as $\mathbb{P}(X_3 = 2) = \sum_{i=1}^3 P_{i2}^{(3)} p_i = (0.209)(1/3) + (0.356)(1/3) + (0.242)(1/3) = 0.269$. The unconditional probability of state 3 may be obtained by similar calculations $\mathbb{P}(X_3 = 3) = (0.258)(1/3) + (0.324)(1/3) + (0.273)(1/3) = 0.285$, or by subtraction from 1, $\mathbb{P}(X_3 = 3) = 1 - 0.446 - 0.269 = 0.285$.

These calculations may be written in the matrix form as follows:

$$(1/3, 1/3, 1/3) \begin{bmatrix} 0.533 & 0.209 & 0.258 \\ 0.320 & 0.356 & 0.324 \\ 0.485 & 0.242 & 0.273 \end{bmatrix} = (0.446, 0.269, 0.285). \quad \square$$

1.4 Classification of States

Consider a discrete-time Markov chain $\{X_n, n \geq 0\}$. We say that starting in state i , the chain *ever enters state j* if for some $n \geq 0$, $X_n = j$, that is,

$$\{\text{chain ever enters state } j \mid \text{it starts in state } i\} = \bigcup_{n=0}^{\infty} \{X_n = j \mid X_0 = i\}.$$

A state j is called *accessible* from state i if, starting in i , the chain will ever enter state j with a positive probability. To denote that state j is accessible from state i , we write $i \rightarrow j$.

Further, if two states are accessible from each other, we say that they *communicate*, and denote it as $i \leftrightarrow j$. Communication property is an equivalence relation. Indeed, (i) it is reflexive since any state communicates with itself in 0 steps, (ii) it is symmetric by definition: if state i communicates with state j , then state j communicates with state i , and (iii) it is transitive because if state i communicates with state j , and state j communicates with state k , then state i communicates with state k .

Now, since communication is an equivalence relation, it means that communication is a *class property*: all states that communicate with each other belong to the same class, and the state space of a Markov chain may be partitioned into classes. If there is only one class, the chain is termed *irreducible*.

Next, a state is called *recurrent* if with probability 1 the chain ever reenters that state. Otherwise, the state is called *transient*. Any state is either recurrent

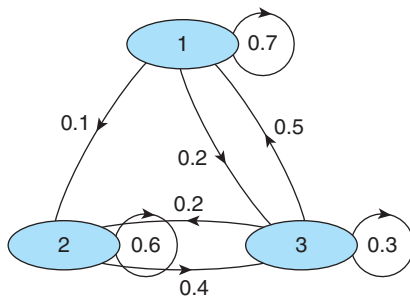
or transient. Furthermore, at least one state of a finite-state Markov chain must be recurrent because if all states are transient, after the chain leaves all the states it has no state to go to. Similar to the communication relation, recurrence and transience are class properties, and the entire class is called *recurrent* or *transient*.

In addition, a state is called *absorbing* if the chain cannot leave it once it enters it. An *absorbing Markov chain* has at least one absorbing state. A state is termed *reflecting* if once the chain leaves it, it cannot return to it.

Finally, the *period* d of a state i is the number such that, starting in i , the chain can return to i only in the number of steps that are multiples of d . A state with period $d = 1$ is called *aperiodic*. Periodicity is a class property.

For a reflecting state, the period is infinite, since the chain never comes back to this state. Absorbing states necessarily have loops and thus are aperiodic states.

EXAMPLE 1.4. Referring back to Example 1.1, we first draw a diagram of the Markov chain. The R code that produces the diagram will be displayed in Example 1.8.

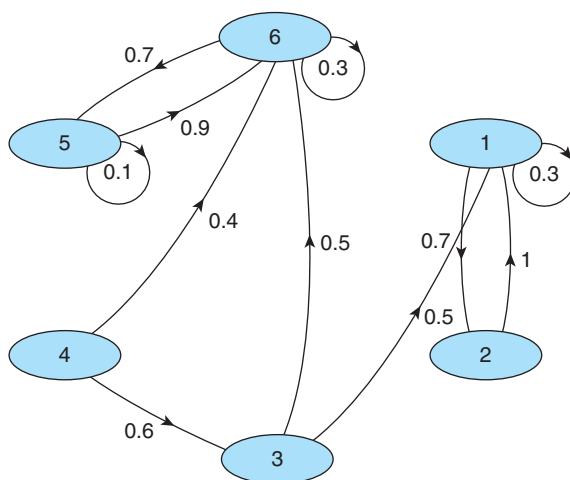


As seen in the diagram, state 2 can be reached from state 1 directly, and states 2 and 3 are directly accessible from each other. Also, state 1 is accessible from state 2 through state 3. Therefore, all three states communicate, and thus the chain has a single class and is irreducible. Since there is only one class, it must be recurrent. Also, the chain has loops for every state, thus it can return to every state in one step and is aperiodic. \square

EXAMPLE 1.5. Consider a Markov chain with the state space $\{1, 2, 3, 4, 5, 6\}$ and the one-step transition probability matrix

$$\begin{array}{c}
 \begin{array}{cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
 \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \left[\begin{array}{cccccc}
 0.3 & 0.7 & 0 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 & 0 & 0 \\
 0.5 & 0 & 0 & 0 & 0 & 0.5 \\
 0 & 0 & 0.6 & 0 & 0 & 0.4 \\
 0 & 0 & 0 & 0 & 0.1 & 0.9 \\
 0 & 0 & 0 & 0 & 0.7 & 0.3
 \end{array} \right]
 \end{array}
 \end{array}$$

The diagram for this chain is given below. The corresponding R code will be presented in Example 1.9.



We can see that states 3 and 4 are reflecting and therefore transient. The chain will transition out of both states and will enter the recurrent class $\{1, 2\}$ or $\{5, 6\}$. States 3 and 4 have infinite periods whereas both recurrent classes are aperiodic due to the existence of the loops. \square

1.5 Limiting Probabilities

In a Markov chain, the *limiting probability* $\pi_j = \lim_{n \rightarrow \infty} P_{ij}^n$ doesn't depend on the initial state i and can be interpreted as the long-run proportion of time that the Markov chain spends in state j . Limiting probabilities are also termed

the *limiting distribution* or *stationary distribution* or *steady-state distribution*. If the stationary distribution exists, it satisfies the system of equations:

$$\pi_j = \sum_{i=0}^{\infty} \pi_i P_{ij}. \quad (1.2)$$

To see this, we condition on the state at time n and write $\mathbb{P}(X_{n+1} = j) = \sum_{i=0}^{\infty} \mathbb{P}(X_{n+1} = j | X_n = i) \mathbb{P}(X_n = i) = \sum_{i=0}^{\infty} P_{ij} \mathbb{P}(X_n = i)$. Letting $n \rightarrow \infty$, and assuming that we can justify exchanging the limit and the summation sign, we get (1.2).

If we combine the limiting probabilities into a row vector $\pi = (\pi_1, \pi_2, \dots)$, then the system of equations (1.2) has the matrix form $\pi = \pi \cdot \mathbf{P}$ where \mathbf{P} is the one-step transition probability matrix. Since the rows of the transition probability matrix add up to 1, the equations in the system are linearly dependent. Nonetheless, we can find all limiting probabilities if we take into account the fact that they must sum up to 1: $\pi_1 + \pi_2 + \dots = 1$.

A Markov chain that has a unique stationary distribution is referred to as *ergodic*.

EXAMPLE 1.6. The stationary distribution for the Markov chain considered in Examples 1.1 – 1.4 solves

$$(\pi_1, \pi_2, \pi_3) = (\pi_1, \pi_2, \pi_3) \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}, \text{ and } \pi_1 + \pi_2 + \pi_3 = 1.$$

Or, equivalently,

$$\begin{cases} \pi_1 = 0.7\pi_1 + 0.5\pi_3 \\ \pi_2 = 0.1\pi_1 + 0.6\pi_2 + 0.2\pi_3, \\ \pi_1 + \pi_2 + \pi_3 = 1, \end{cases} \quad \text{or} \quad \begin{cases} \pi_3 = 0.6\pi_1 \\ \pi_2 = 0.55\pi_1 \\ 2.15\pi_1 = 1, \end{cases}$$

resulting in the solution $\pi_1 = 0.4651$, $\pi_2 = 0.2558$, and $\pi_3 = 0.2791$. It means that in a long run, the chain spends roughly 46.5% of the time in state 1, 25.6% of the time in state 2, and 27.9% of the time in state 3. Since the stationary distribution is unique, it is an ergodic chain. \square

EXAMPLE 1.7. As will be demonstrated in Example 1.9, for the Markov chain in Example 1.5, the system of equations (1.2) has two solutions

$$(0.5882, 0.4118, 0, 0, 0, 0) \text{ and } (0, 0, 0, 0, 0.4375, 0.5625).$$

This happens because the chain contains two recurrent classes, $\{1, 2\}$ and $\{5, 6\}$. Since there are two solutions, the chain is *non-ergodic*. Denote these two vectors by π_1 and π_2 , respectively. Then any linear combination of the

form $\alpha\pi_1 + (1 - \alpha)\pi_2$ where $0 < \alpha < 1$ is a solution, and therefore, there actually exist infinitely many solutions. Any such solution is called an *invariant measure*. Neither is considered to be the stationary distribution. \square

1.6 Computations in R

To mimic in R the analysis done in examples in the previous section, three packages are required: “diagram” to plot the diagram, “expm” to exponentiate matrices, and “markovchain” to determine recurrent and transient classes, absorbing states, and the steady-state distribution. The script is as follows:

```
install.packages("diagram")
install.packages("expm")
install.packages("markovchain")
```

- First, an $n \times n$ transition probability matrix should be specified:

```
tm.name<- matrix(c(p11.value, p12.value,...,pnn.value), nrow=n.value,
ncol=n.value, byrow=TRUE)
```

- To plot the diagram with the arrows pointing in the correct direction, the transition matrix must be transposed. This can be done by typing

```
tr.tm.name<- t(tm.name)
```

The diagram is drawn using `plotmat` function. The graph depicts state names inside boxes and directed lines connecting the boxes labeled by the corresponding transition probabilities. The syntax is:

```
library(diagram)
plotmat(tr.tm.name, <arguments>)
```

The arguments in the above function are:

- `pos`, a vector specifying the number of boxes in each row in the diagram. For instance, in a three-state chain, `pos=c(1,2)` means that state 1 is depicted in the top row, and states 2 and 3 are in the next row. By default, boxes are plotted in a circle.
- `name=c("state1.name", "state2.name", ...)`, the list of state names. By default, natural numbers are used.
- `arr.col`, color of inside of all arrowheads (excluding the contours). Black by default.

- `arr.lcol`, color of all arrow lines. Black by default.
- `arr.length`, length of all arrows.
- `arr.pos`, relative position of arrowheads on lines (excluding loops), a value between 0 and 1. By default, arrows are positioned in the middle.
- `arr.type`, type of arrowhead, some options are `curved`, `triangle`, or `simple`.
- `arr.width`, width of all arrows.
- `box.cex`, size of labels in boxes (i.e., state names). The magnitude is relative to the default value of character expansion.
- `box.col`, color of inside of all boxes (excluding contours). By default, the color is white.
- `box.lcol`, color of contours of all boxes. Black by default.
- `box.lwd`, width of contours of all boxes.
- `box.prop`, ratio of length over width of all boxes. The ratio is equal to 1 by default.
- `box.size`, size of all boxes.
- `box.type`, type of all boxes, some options are `rect`, `ellipse`, `round` (a rectangle with rounded edges), `circle`, `diamond`, `hexa` (a hexagonal shape).
- `cex.txt`, size of labels next to arrows (i.e., values of respective probabilities).
- `lcol`, color of all lines, contours of all arrows, and contours of all boxes. Black by default.
- `lwd`, width of all arrow lines, excluding loops.
- `self.cex`, size of all loops (also termed “self-arrows”).
- `self.arrpos`, angle in radians of arrow positions on all loops relative to the x -axis.
- `self.lwd`, width of all loops.
- `self.shiftx`, shift of all loops along the x -axis.
- `self.shifty` shift of all loops along the y -axis.
- `txt.col`, color of labels in boxes (i.e., state names).

- To compute an n -step transition probability matrix for a specific value of n , use the following code:

```
library(expm)
nstepmatrix.name<- tm.name% ^ %n.value
```

- Given the initial distribution, the lines below calculate the unconditional distribution after n steps.

```
init.p.name <- c(p1.init.value,p2.init.value,...)
uncond.dist.name<- init.p.name%*%nstepmatrix.name
```

- To determine recurrent and transient classes, absorbing states, and the limiting distribution, a discrete-time Markov chain must be created as an object. It can be done as follows:

```
library(markovchain)
dtmc.name<- new("markovchain", transitionMatrix=tm.name,
states=c("state1.name", "state2.name",...))
```

Then state characteristics may be obtained as

```
recurrentClasses(dtmc.name)
transientClasses(dtmc.name)
absorbingStates(dtmc.name)
steadyStates(dtmc.name)
```

Note that R doesn't identify reflecting states.

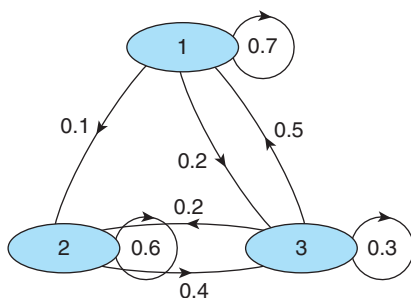
In R, one can compute the period of an irreducible (a single-class) Markov chain only. The syntax is `period(dtmc.name)`.

EXAMPLE 1.8. The results obtained for the Markov chain from Examples 1.1-1.4 and 1.6 can be produced in R as presented below.

```
#specifying transition probability matrix
tm<- matrix(c(0.7, 0.1, 0.2, 0.0, 0.6, 0.4, 0.5, 0.2, 0.3),
nrow=3, ncol=3, byrow=TRUE)

#transposing transition probability matrix
tm.tr<- t(tm)

#plotting diagram
library(diagram)
plotmat(tm.tr, pos=c(1,2), arr.length=0.3, arr.width=0.1,
box.col="light blue", box.lwd=1, box.prop=0.5, box.size=0.12,
box.type="circle", cex.txt=0.8, lwd=1, self.cex=0.6,
self.shiftx=0.17, self.shifty=-0.01)
```



```
#computing three-step transition probability matrix
library(expm)
print(tm3<- tm% ^ % 3)
```

```
      [,1] [,2] [,3]
[1,] 0.533 0.209 0.258
[2,] 0.320 0.356 0.324
[3,] 0.485 0.242 0.273
```

```
#computing unconditional distribution after three steps
init.p<- c(1/3, 1/3, 1/3)
init.p%* %tm3
```

```
      [,1] [,2] [,3]
[1,] 0.446 0.269 0.285
```

```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1",
"2", "3"))
```

```
#computing Markov chain characteristics
recurrentClasses(mc)
```

```
"1" "2" "3"
```

```
transientClasses(mc)
```

```
list()
```

```
absorbingStates(mc)
```

```
character(0)
```

```
period(mc)
```

```
1
```

```
round(steadyStates(mc), digits=4)
```

```
      1      2      3
0.4651 0.2558 0.2791
```

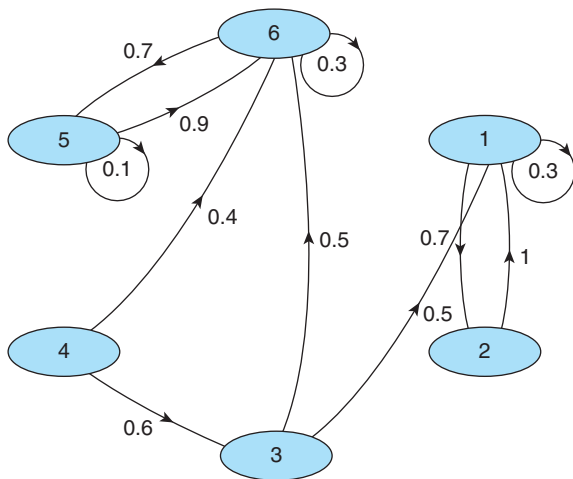
□

EXAMPLE 1.9. Consider the Markov chain from Examples 1.5 and 1.7. We run the following R code to produce the diagram, invariant vectors, and to verify the state classification.

```
#specifying transition probability matrix
tm<- matrix(c(0.3,0.7,0,0,0,0,1,0,0,0,0,0,0.5,0,0,0,0,0.5,
0,0,0.6,0,0,0.4,0,0,0,0,0.1,0.9,0,0,0,0,0.7,0.3), nrow=6,
ncol=6, byrow=TRUE)

#transposing transition probability matrix
tm.tr<- t(tm)

#plotting diagram
library(diagram)
plotmat(tm.tr, arr.length=0.3, arr.width=0.1, box.col="light
blue", box.lwd=1, box.prop=0.5, box.size=0.09,
box.type="circle", cex.txt=0.8, lwd=1, self.cex=0.3,
self.arrpos=0.3, self.shiftx=0.09, self.shifty=-0.05)
```



```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1",
"2", "3", "4", "5", "6"))

#computing Markov chain characteristics
recurrentClasses(mc)
```

"1" "2"

"5" "6"


```
transientClasses(mc)
```

```
"3"
```

```
"4"
```

```
#finding periods of irreducible Markov chains
tm12.ir<- matrix(c(0.3,0.7,1,0), nrow=2, ncol=2, byrow=TRUE)
mc12.ir<- new("markovchain", transitionMatrix=tm12.ir,
states=c("1","2"))
period(mc12.ir)
```

```
1
```

```
tm56.ir<- matrix(c(0.1,0.9,0.7,0.3), nrow=2, ncol=2,
byrow=TRUE)
mc56.ir<- new("markovchain", transitionMatrix=tm56.ir,
states=c("5","6"))
period(mc56.ir)
```

```
1
```

```
#finding steady-state distribution
round(steadyStates(mc), digits=4)
```

```
      1      2  3  4      5      6
0.0000 0.0000 0  0  0.4375 0.5625
0.5882 0.4118 0  0  0.0000 0.0000
```

```
□
```

1.7 Simulations in R

In R, to simulate simultaneously *ntraj.value* trajectories of a k -state Markov chain over *nsteps.name* time-steps, the function `rmarkovchain()` in the library `markovchain` can be used. It keeps track of the step number and the Markov chain state at that step. Each new state is chosen according to a multinomial probability distribution with the probability mass function

$$P(x_1, \dots, x_k, p_1, \dots, p_k) = \mathbb{P}(X_1 = x_1, \dots, X_k = x_k) \\ = \frac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}, \text{ where } x_1 + \cdots + x_k = n.$$

Given the initial vector of probabilities (p_1^1, \dots, p_k^1) , the multivariate probability distribution is found recursively:

$$(p_1^n, p_2^n, \dots, p_k^n) = (p_1^{n-1}, p_2^{n-1}, \dots, p_k^{n-1}) \mathbf{P}, \quad n = 2, 3, \dots, \quad (1.3)$$

where \mathbf{P} denotes the one-step transition probability matrix of the Markov chain.

Assuming that the Markov chain object *mc.name* has already been created, the syntax is as follows:

```
#specifying total number of trajectories
ntraj.name<- ntraj.value

#specifying total number of steps
nsteps.name<- nsteps.value

#specifying initial probability
init.prob.name<- c(p1.value, p2.value, ..., pk.value)

#specifying matrix containing states
states.matrix.name<- matrix(NA, nrow=nsteps.name, ncol=ntraj.name)

#specifying seed
set.seed(value)

#generating initial state
init.state.name<- sample(1:k, 1, prob=init.prob.name)

#simulating states
for (i in 1:ntraj.name)
  states.matrix.name[,i]<- rmarkovchain(n=nsteps.name-1,
  object=mc.name,
  t0=init.state.name, include.t0=TRUE)
```

- The value of the seed tells R where to start reading off in the table of random digits. This is done for the reproducibility of results. If the code is run again, it will produce the same trajectories.

- The function `sample(1:k, 1, prob=init.prob.name)` draws one state from among the k states and the sampling is done according to the multinomial distribution with the probability vector *init.prob.name*.

- The final product of the above code is a matrix with dimensions *nsteps.value* by *ntraj.value* with columns containing state names for individual trajectories.

As an alternative to using the built-in function `rmarkovchain()`, one can create a user-defined function that simulates trajectories. The syntax is

```
function.name<- function(tm.name, init.prob.name, nsteps.name) {
  states.name<- numeric(nsteps.name)
  states.name[1]<- sample(1:k, 1, prob=init.prob.name)

  for(t in 2:nsteps.name) {
    prob.name<- tm.name[states.name[t-1],]
    states.name[t]<- sample(1:k, 1, prob=prob.name)
  }
  return(states.name)
}
```

```
set.seed(value)
states.matrix.name<- matrix(NA, nrow=nsteps.name, ncol=ntraj.name)
  for (j in 1:ntraj.name)
    states.matrix.name<- function.name(tm.name, init.prob.name,
      nsteps.name)
```

Finally, the *ntraj.value* trajectories can be overlayed on a single graph by means of the `matplot()` function with the following syntax:

```
#plotting simulated trajectories
matplot(states.matrix.name, <arguments>)
```

where the arguments are: `type="l"` to connect the dots, `lty=1` to draw a solid line, `lwd=2` to make the lines appear thicker, `col=a:b` to choose a sequential subset *a* to *b* of *ntraj.name* colors from this cyclic list {1=black, 2=red, 3=green, 4=dark blue, 5=sky blue, 6=pink, 7=yellow, 8=gray, 9=black, 10=red, 11=green, etc.}, `xlim=c(x.lower.value, x.upper.value)` to limit the *x*-axis, if needed, `ylim=c(y.lower.value, y.upper.value)` to limit the *y*-axis, if needed, `xlab="step.name"` and `ylab="state.name"` to display *step.name* on the *x*-axis and *state.name* on the *y*-axis. Colors can also be specified as an explicit list: `col=c("red", "green", "blue")`, for example. To add grid lines to the coordinate system on the plot, the argument `panel.first=grid()` is used.

To change tick marks on either axis and to modify labels for ticks, one can remove axes by including in `matplot()` the arguments `xaxt="n"` and `yaxt="n"`, and then specify the new axes as `axis(side=number, at=range for ticks)`, where `side=1` is for *x*-axis, and `side=2` is for *y*-axis.

To enhance the clarity of what states the depicted trajectories are at, dots can be added at every step. To do so, use `points(x values, y values, pch=16)`, where the argument `pch=16` identifies the plotting character for the points as a filled circle.

Additionally, to visualize convergence of unconditional probabilities p_n , computed recursively in accordance with (1.3), the following R syntax can be implemented. We assume that there are k states in the Markov chain, and the transition probability matrix has been defined as *tm.name* object.

```
#specifying total number of steps
nsteps.name<- nsteps.value

#specifying matrix containing probabilities
probs.matrix.name<- matrix(NA, nrow=nsteps.name, ncol=k)

#specifying initial probability
init.prob.name<- c(p1.value, p2.value, ..., pk.value)

#computing unconditional probabilities
probs.matrix.name[1,] <- init.prob.name
  for(n in 2:nsteps.name)
    probs.matrix.name[n,]<- probs.matrix.name[n-1,]%*%tm.name

#plotting probabilities against steps by state
matplot(probs.matrix.name, type="l", lty=1, col=a:b,
ylim=c(lower.value, upper.value), ylab="probability", xlab="step")
```

In addition, a legend should be added to match the lines with states' numbers.

```
legend(legend.position, c("state 1", "state 2", ..., "state k"),
lty=1, col=a:b)
```

- The choices for `legend.position` on the graph are: "topright", "right", "bottomright", "top", "center", "bottom", "topleft", "left", and "bottomleft".

SIMULATION 1.1. Consider the Markov chain discussed in Examples 1.1-1.4, 1.6, and 1.8. Recall that the transition probability matrix for this chain is specified as

$$\begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.0 & 0.6 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}.$$

Below we simulate two trajectories of this chain with the initial state chosen at random, that is, with the probability vector $(1/3, 1/3, 1/3)$. First, we demonstrate how to utilize the built-in function `rmarkovchain()`.

```
#specifying transition probability matrix
tm<- matrix(c(0.7, 0.1, 0.2, 0.0, 0.6, 0.4, 0.5, 0.2, 0.3),
nrow=3, ncol=3, byrow=TRUE)

#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1",
"2", "3"))

#specifying total number of steps
nsteps<- 25

#specifying initial probability
p0<- c(1/3, 1/3, 1/3)

#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=2)

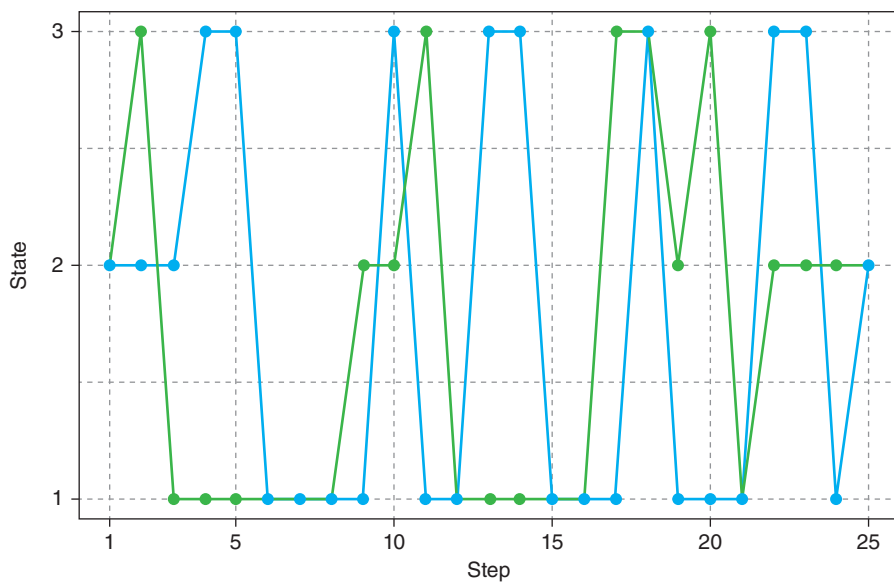
#specifying seed
set.seed(2443927)

#simulating trajectories
for (i in 1:2)
state0<- sample(1:3, 1, prob=p0)
MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc,
t0=state0, include.t0=TRUE)

#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=3:4,
xaxt="n", yaxt="n", ylim=c(1,3), xlab="Step", ylab="State",
panel.first=grid())

axis(side=1, at=c(1,5,10,15,20,25))
axis(side=2, at=1:3)

points(1:nsteps, MC.states[,1], pch=16, col=3)
points(1:nsteps, MC.states[,2], pch=16, col=4)
```



Now we present the code with a user-defined function that simulates trajectories, bypassing the built-in `rmarkovchain()` function. We use the same seed as above and obtain the same trajectories. The code and graph follow.

```
#creating user-defined function
MC<- function(tm, p0, nsteps) {
  states<- numeric(nsteps)
  states[1]<- sample(1,1,prob=p0)

  for(t in 2:nsteps) {
    p<- tm[states[t-1],]
    states[t]<- sample(1,1,prob=p)
  }

  return(states)
}

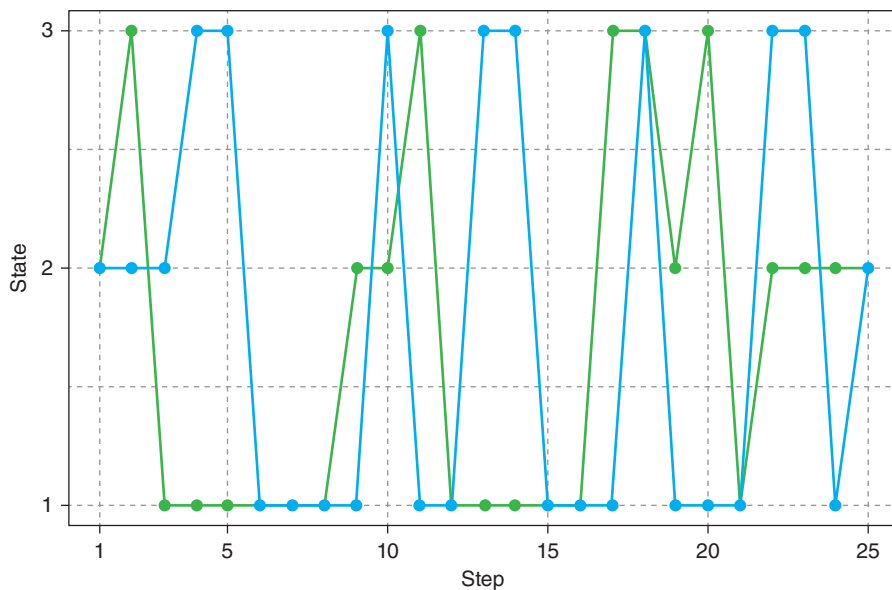
#specifying seed
set.seed(2443927)

#simulating trajectories
MC.states2<- matrix(NA, nrow=nsteps, ncol=2)
for (j in 1:2)
  MC.states2[,j]<- MC(tm, p0, nsteps)
```

```
#plotting simulated trajectories
matplot(MC.states2, type="l", lty=1, lwd=2, col=3:4,
xaxt="n", yaxt="n", ylim=c(1,3), xlab="Step", ylab="State",
panel.first= grid())

axis(side=1, at=c(1,5,10,15,20,25))
axis(side=2, at=c(1,2,3))

points(1:nsteps, MC.states2[,1], pch=16, col=3)
points(1:nsteps, MC.states2[,2], pch=16, col=4)
```



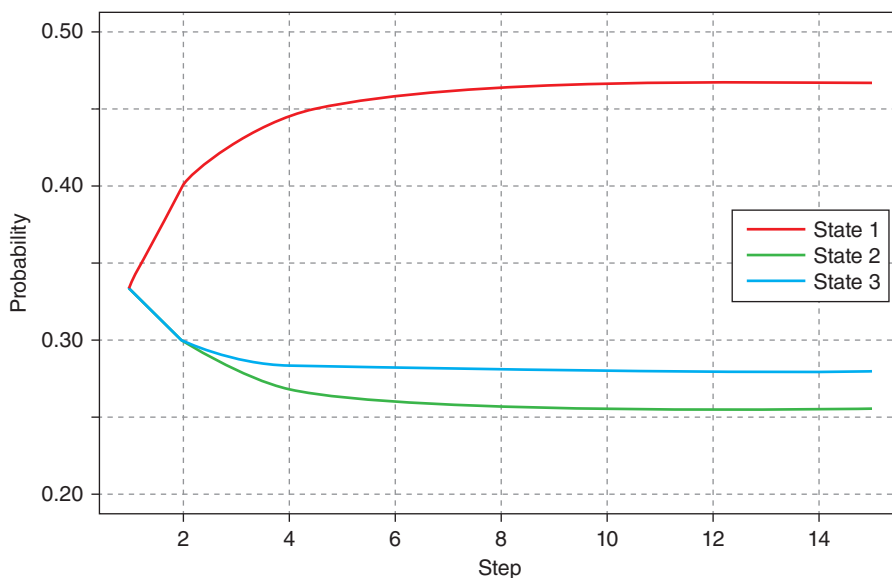
Now we compute iteratively probability vectors and plot them against the step number. The code and plot are given below.

```
#specifying matrix containing probabilities
probs<- matrix(NA, nrow=nsteps, ncol=3)

#specifying total number of steps
nsteps<- 15
```

```
#computing probabilities p_n
probs[1,] <- p0
for(n in 2:nsteps)
  probs[n,]<- probs[n-1,]%*%tm

#plotting probabilities against steps by state
matplot(probs, type="l", lty=1, lwd=2, col=2:4,
        ylim=c(0.2,0.5), xlab="Step ", ylab="Probability",
        panel.first=grid())
legend("right", c("State 1 ", "State 2", "State 3"), lty=1,
       lwd=2, col=2:4)
```



We also output the values of the probabilities to see that they converge to the steady state (0.4651, 0.2558, 0.2791) on step 14.

```
> probs
      [,1]      [,2]      [,3]
<lines omitted>
[13,] 0.4650353 0.2558698 0.2790949
[14,] 0.4650721 0.2558444 0.2790835
```

□

SIMULATION 1.2. We return to the Markov chain studied in Examples 1.5, 1.7, and 1.9. First, we use the already-created object `mc` to generate three trajectories that start at a randomly chosen state. In the code that follows, we apply the `rmarkovchain()` function to simulate the trajectories.

```
#specifying total number of steps
nsteps<- 20

#specifying initial probability
p0<- c(1/6, 1/6, 1/6, 1/6, 1/6, 1/6)

#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=3)

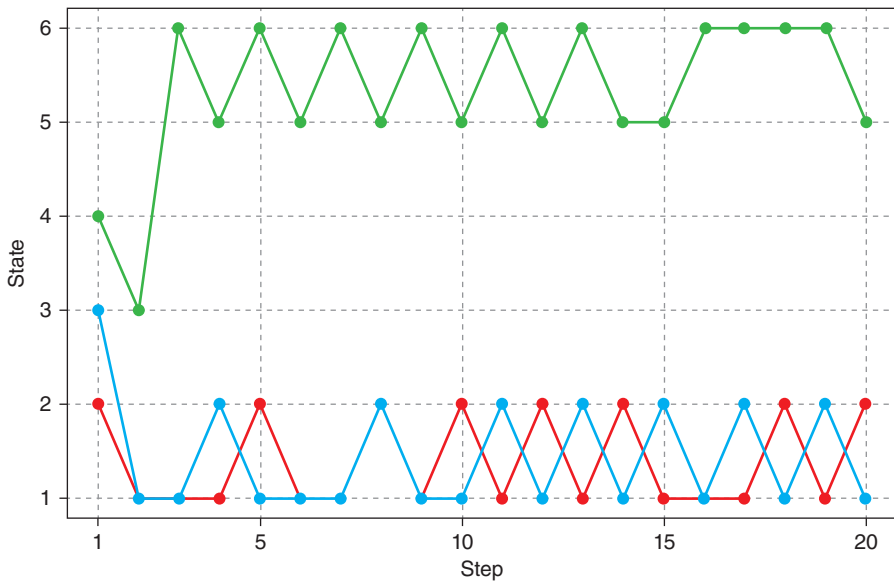
#specifying seed
set.seed(765881)

#simulating trajectories
for (i in 1:3) {
  state0<- sample(1:6, 1, prob=p0)
  MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc,
    t0=state0,
    include.t0=TRUE)
}

#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=2:4, xaxt="n",
  ylim=c(1,6), xlab="Step", ylab="State", panel.first=grid())

axis(side=1, at=c(1,5,10,15,20))

points(1:nsteps, MC.states[,1], pch=16, col=2)
points(1:nsteps, MC.states[,2], pch=16, col=3)
points(1:nsteps, MC.states[,3], pch=16, col=4)
```



Next, we compute and plot the unconditional probabilities p_n against n . As n increases, these probabilities converge to an invariant probability measure, which heavily depends on the initial state of the Markov chain. These limiting values will be the same for states 1 and 2 since they belong to the same class, for states 5 and 6 for the same reason, and separately for state 3, and for state 4. From the theoretical viewpoint, the invariant measure for states 1 and 2 is $(0.5882, 0.4118, 0, 0, 0, 0)$. For states 5 and 6, it is $(0, 0, 0, 0, 0.4375, 0.5625)$. From state 3 the chain is equally likely to enter either $\{1, 2\}$ or $\{5, 6\}$, therefore, the invariant measure is found as $(0.5)(0.5882, 0.4118, 0, 0, 0, 0) + (0.5)(0, 0, 0, 0, 0.4375, 0.5625) = (0.2941, 0.2059, 0, 0, 0.21875, 0.28125)$. From state 4, the chain will enter the class $\{5, 6\}$ directly with probability 0.4, or through state 3, with probability $(0.6)(0.5) = 0.3$, hence, it enters class $\{5, 6\}$ with the total probability $0.4 + 0.3 = 0.7$, and enters class $\{1, 2\}$ only through state 3 with the probability $(0.6)(0.5) = 0.3$. As a result, the invariant measure for state 4 is $(0.3)(0.5882, 0.4118, 0, 0, 0, 0) + (0.7)(0, 0, 0, 0, 0.4375, 0.5625) = (0.17646, 0.12354, 0, 0, 0.30625, 0.39375)$.

We run the code below six times, every time choosing a different initial state. As the output, we present the graphs for each initial state, and print the vector of probabilities for steps 28 through 30.

```
#specifying total number of steps
nsteps<- 30
```

```

#specifying initial state distribution (state 1)
p0<- c(1,0,0,0,0,0)
#(state 2) p0<- c(0,1,0,0,0,0), (state 3) p0<-
c(0,0,1,0,0,0),
#(state 4) p0<- c(0,0,0,1,0,0), (state 5) p0<-
c(0,0,0,0,1,0),
#(state 6) p0<- c(0,0,0,0,0,1)

#specifying matrix containing probabilities
probs<- matrix(NA, nrow=nsteps, ncol=6)

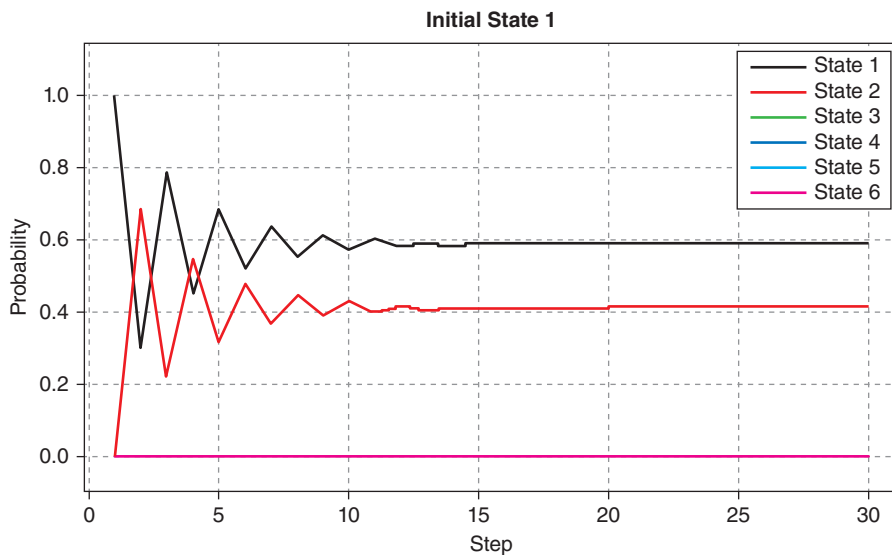
#computing probabilities p_n
probs[1,] <- p0

for(n in 2:nsteps)
probs[n,]<- probs[n-1,]%*%tm

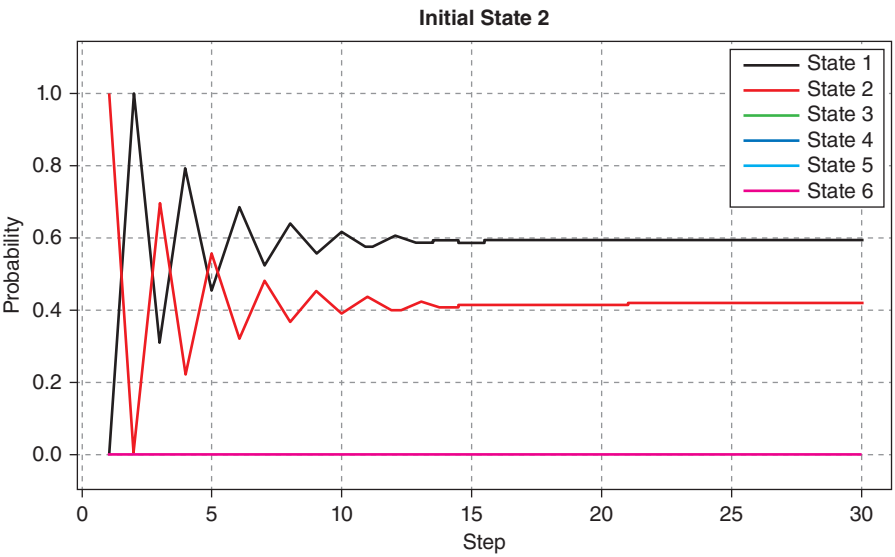
#plotting probabilities vs. step by state
matplot(probs, main="Initial State 1", type="l", lty=1,
lwd=2, col=1:6, ylim=c(-0.05, 1.1), xlab="Step",
ylab="Probability", panel.first = grid())

legend("topright", c("State 1", "State 2", "State 3", "State
4", "State 5", "State 6"), lty=1, lwd=2, col=1:6)

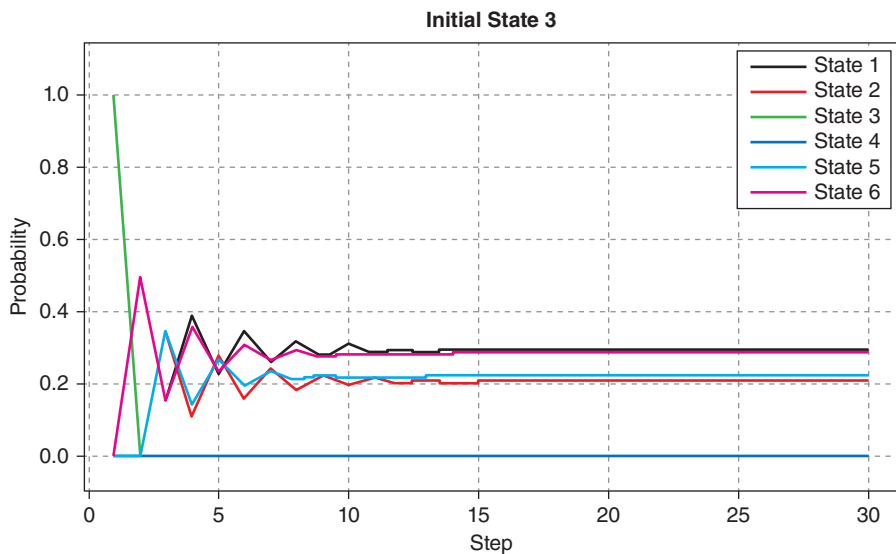
```



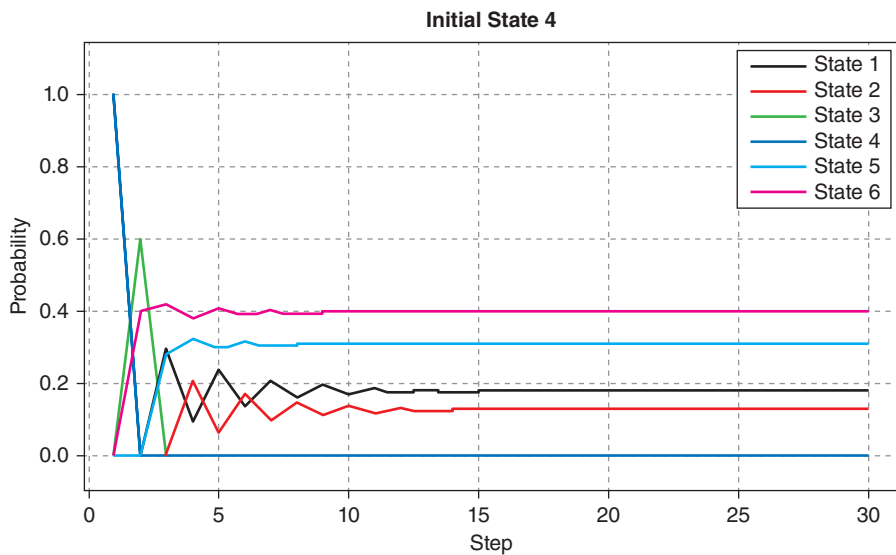
	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
	<lines omitted>					
[28,]	0.5882082	0.4117918	0	0	0	0
[29,]	0.5882542	0.4117458	0	0	0	0
[30,]	0.5882220	0.4117780	0	0	0	0



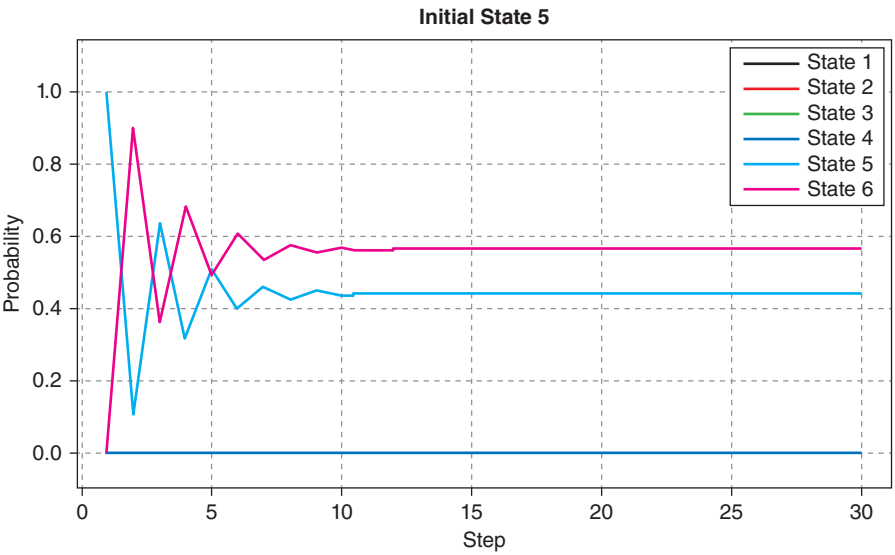
	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
	<lines omitted>					
[28,]	0.5882739	0.4117261	0	0	0	0
[29,]	0.5882082	0.4117918	0	0	0	0
[30,]	0.5882542	0.4117458	0	0	0	0



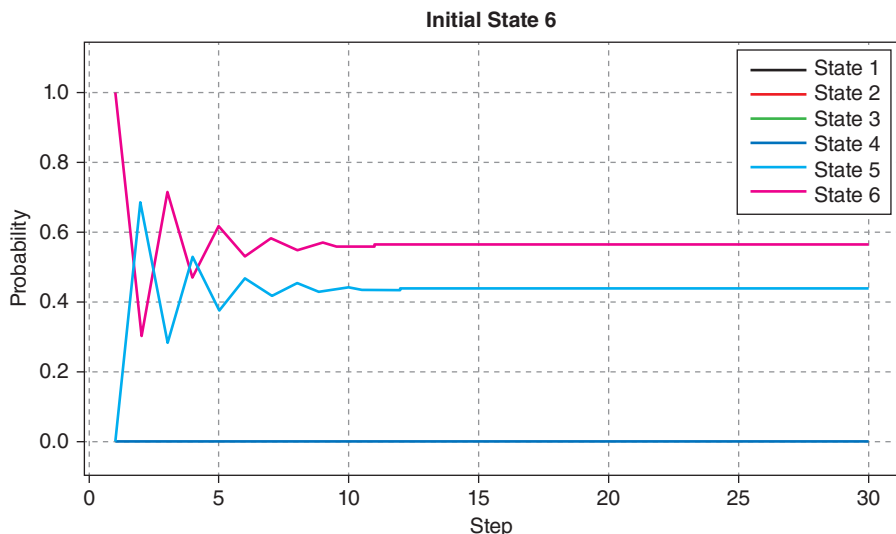
	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
	<lines omitted>					
[28,]	0.2941370	0.2058630	0	0	0.2187496	0.2812504
[29,]	0.2941041	0.2058959	0	0	0.2187502	0.2812498
[30,]	0.2941271	0.2058729	0	0	0.2187499	0.2812501



	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
	<lines omitted>					
[28,]	0.1764540	0.1235460	0	0	0.3062501	0.3937499
[29,]	0.1764822	0.1235178	0	0	0.3062500	0.3937500
[30,]	0.1764625	0.1235375	0	0	0.3062500	0.3937500



	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
	<lines omitted>					
[28,]	0	0	0	0	0.4374994	0.5625006
[29,]	0	0	0	0	0.4375003	0.5624997
[30,]	0	0	0	0	0.4374998	0.5625002



	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
	<lines omitted>					
[28,]	0	0	0	0	0.4374994	0.5625006
[29,]	0	0	0	0	0.4375003	0.5624997
[30,]	0	0	0	0	0.4374998	0.5625002

□

1.8 Applications of Markov Chain

APPLICATION 1.1. In 1913, A. A. Markov (1856-1922), a Russian mathematician after whom Markov chains are named, published an article where he analyzed sequences of vowels and consonants among the first 20,000 letters of “Eugene Onegin” by A. S. Pushkin.² Two silent letters in the Russian language that are indicators of the softness of a preceding sound and are neither vowels nor consonants were ignored. He cleverly argued that the appearance of vowels and consonants are remarkably dependent in such a way that this literary work may be approximated by what we now call a Markov chain. He went ahead and computed the one-step transition probability matrix for this chain with the state space $S = \{\text{vowel (v)}, \text{consonant (c)}\}$. He calculated that

²Markov, A. A. (1913). “An example of statistical investigation of the text Eugene Onegin concerning the connection of samples in chains.” (In Russian). *Bulletin of the Imperial Academy of Sciences of St. Petersburg*, 7(3): 153 – 162.

there are 8,638 vowels in the text (respectively, 11,362 consonants) and 1,104 vowel-vowel sequences. It is also important to notice that the text starts with a consonant and ends with a vowel, meaning that nothing transitions into the first consonant and the total number of transitions into vowels adds up to the total number of vowels. Thus, there must be $8,638 - 1,104 = 7,534$ consonant-vowel sequences, $11,362 - 7,534 = 3,828$ consonant-consonant sequences, and $11,361 - 3,828 = 7,533$ vowel-consonant sequences. The transition probability matrix can then be computed as

$$\mathbf{P} = \begin{array}{c} \begin{array}{cc} & \begin{array}{c} \text{v} \\ \text{c} \end{array} \\ \begin{array}{c} \text{v} \\ \text{c} \end{array} & \begin{bmatrix} \frac{1104}{8637} = 0.1278 & \frac{7533}{8637} = 0.8722 \\ \frac{7534}{11362} = 0.6631 & \frac{3828}{11362} = 0.3369 \end{bmatrix} \end{array} \end{array}.$$

Now, out of curiosity, we compute the limiting probabilities, which, theoretically speaking, represent proportions of vowels and consonants in the text. Since we know the exact numbers, we compute $\pi_v = \frac{8638}{20000} = 0.4319$, and $\pi_c = 1 - 0.4319 = 0.5681$. Now, resorting to R, we obtain the same values:

```
#specifying the transition probability matrix
tm<- matrix(c(0.1278, 0.8722, 0.6631, 0.3369), nrow=2,
ncol=2,
byrow=TRUE)

#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("v",
"c"))

#computing limiting probabilities
steadyStates(mc)
```

```
      v      c
0.4319026 0.5680974
```

To verify how accurate A. A. Markov was with his estimation of the one-step transition probability matrix, we ran the same analysis on the entire novel. First, we precleaned the text, removing everything besides the Cyrillic letters of the pre-1918 reform Russian alphabet. Then we ran the code presented below that changes the capitalization to lowercase, then removes blanks, line breaks, punctuation marks, and the soft and hard signs of the Russian alphabet, leaving all the cleaned analysis-ready string containing a total of 106,508 characters. To compute the combinations vv , cv , vc , and cc , we shift the string to the left by one position, by inserting a blank in the front and deleting

the last letter to preserve the length of the string. When we line up the lagged string x_1 with the original string x_2 , we obtain the preceding and following letters in the string. After that, we apply simple Boolean logic to calculate the number of vowels and consonants and the number of combinations of all four types. The code and output follow.

```
text <- read_file("./Onegin.txt")

#text cleaning
#gsub() = global substitution function=replaces all instances
lowercase<- tolower(text)
no.blanks<- gsub(" ","",lowercase)
no.line.breaks<- gsub("\r\n", "", no.blanks)
no.punctuation<- gsub("[[:punct:]]","",no.line.breaks)
no.soft.signs<- gsub("ъ", "", no.punctuation)
#removing all hard signs
clean.string<- gsub("Ъ","", no.soft.signs)

#splitting single string into characters
x2<- strsplit(clean.string, "")

#shifting string by one position
no.last<- substr(clean.string, 1, nchar(clean.string)-1)
first.blank<- str_c("", no.last)
x1<- strsplit(first.blank,"")

#Note: In pre-1918 Russian language "ѣ" was considered a
vowel
vowels<-c("а", "е", "ё", "ѣ", "і", "ѥ", "о", "у", "ѡ", "ѣ",
"э", "ю", "я")

consonants<- c("б", "в", "г", "д", "ж", "з", "к", "л", "м",
"н", "п", "р", "с", "т", "ф", "х", "ц", "ч", "ш", "щ", "ѡ")

#computing number of vowels, consonants, and four
combinations
for (counter in 1:nchar(x2)){
v<- ifelse(x2[[counter]] %in% vowels,1,0)
c<- ifelse(x2[[counter]] %in% consonants,1,0)
```

```

vv<- ifelse(x1[[counter]] %in% vowels & x2[[counter]] %in%
vowels,1,0)
vc<- ifelse(x1[[counter]] %in% vowels & x2[[counter]] %in%
consonants,1,0)
cv<- ifelse(x1[[counter]] %in% consonants & x2[[counter]]
%in% vowels,1,0)
cc<- ifelse(x1[[counter]] %in% consonants & x2[[counter]]
%in% consonants,1,0)
}
sum(v)

```

46475

```
sum(c)
```

60033

```
sum(vv)
```

6368

```
sum(vc)
```

40107

```
sum(cv)
```

40107

```
sum(cc)
```

19925

Note that all these numbers add up perfectly. The total number of vowels and consonants is $46,475 + 60,033 = 106,508$. Since the text starts and ends with consonants, all vowels have leading and trailing letters and therefore, we must have $\text{sum}(v) = \text{sum}(vv) + \text{sum}(vc) = \text{sum}(vv) + \text{sum}(cv)$. Indeed, $\text{sum}(vc) = \text{sum}(cv) = 40,107$ and $\text{sum}(vv) + \text{sum}(vc) = 6,368 + 40,107 = 46,475 = \text{sum}(v)$. Also, all consonants but the first one have leading letters and all consonants but the last one have trailing letters. Hence, we must have $\text{sum}(c) - 1 = \text{sum}(cv) + \text{sum}(cc) = \text{sum}(vc) + \text{sum}(cc)$, which indeed holds since $40,107 + 19,925 = 60,032 = 60,033 - 1$.

Turning these numbers into the one-step transition probability matrix, we obtain

$$\mathbf{P} = \begin{array}{c} \begin{array}{cc} & \begin{array}{c} v \\ c \end{array} \\ \begin{array}{c} v \\ c \end{array} & \begin{bmatrix} \frac{6368}{46475} = 0.1370 & \frac{40107}{46475} = 0.8630 \\ \frac{40107}{60032} = 0.6681 & \frac{19925}{60032} = 0.3319 \end{bmatrix} \end{array} \end{array}$$

On the other hand, we estimate

$$\begin{aligned}\widehat{\mathbb{P}}(X_3 = \mathbf{c} \mid X_2 = \mathbf{c}) &= \frac{\widehat{\mathbb{P}}(X_2 = \mathbf{c}, X_3 = \mathbf{c})}{\widehat{\mathbb{P}}(X_2 = \mathbf{c})} \\ &= \frac{\widehat{\mathbb{P}}(\mathbf{ccc}) + \widehat{\mathbb{P}}(\mathbf{vcc})}{\widehat{\mathbb{P}}(\mathbf{ccc}) + \widehat{\mathbb{P}}(\mathbf{ccv}) + \widehat{\mathbb{P}}(\mathbf{vcc}) + \widehat{\mathbb{P}}(\mathbf{vcv})} = \frac{3k + 8k}{3k + 8k + 8k + 3k - 1} \\ &= \frac{11k}{22k - 1} \approx \frac{1}{2}, \text{ for large } k.\end{aligned}$$

Since $\frac{3}{11} \neq \frac{11k}{22k-1}$, this chain doesn't satisfy the definition (1.1), and hence we have shown that this process is not a Markov chain. \square

APPLICATION 1.3. According to the Mendelian model of gene inheritance in humans, named after Gregor Johann Mendel (1822-1884), a specific genetic trait is determined by a pair of genes, that can be of three types: AA , Aa , or aa . During reproduction, an offspring inherits one gene of the pair from each parent, and genes are selected at random, independently of each other.

Suppose gene a is a mutant gene or a person possessing this gene is a carrier of a disease. Consider the genotype of the offspring in successive generations if the second parent always has genotype AA . This can be presented as a Markov chain with the state space $S = \{AA, Aa, aa\}$ and transition probability matrix

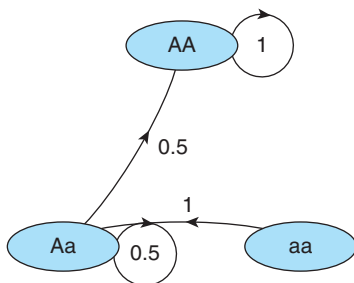
$$\mathbf{P} = \begin{array}{cc} & \begin{array}{ccc} AA & Aa & aa \end{array} \\ \begin{array}{c} AA \\ Aa \\ aa \end{array} & \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 0.5 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array}.$$

Indeed, if parents have genes (AA, AA) , then their offspring are bound to have genes AA with probability 1. If parents have genes (Aa, AA) , they are equally likely to spit into AA or Aa , and finally, if parents have genes (aa, AA) , their offspring with certainty will have genes Aa . The diagram for this model is

```
#specifying the transition probability matrix
tm<- matrix(c(1, 0, 0, 0.5, 0.5, 0, 0, 1, 0), nrow=3, ncol=3,
byrow=TRUE)

#transposing the transition probability matrix
tm.tr<- t(tm)
```

```
#plotting the diagram for the Markov chain
library(diagram)
plotmat(tm.tr, pos=c(1,2), name=c("AA", "Aa", "aa"),
arr.length=0.3, arr.width=0.1, box.col="light blue",
box.lwd=1, box.prop=0.5, box.size=0.12, box.type="circle",
cex.txt=0.8, lwd=1, self.cex=0.6, self.shiftx=0.17,
self.shifty=-0.01)
```



Note that states *Aa* and *aa* are transient states, and *AA* is the absorbing state. It means that in a long run, the gene *a* will disappear from the population. To convince ourselves, we compute the stationary distribution.

```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm,
states=c("AA", "Aa", "aa"))

#computing stationary distribution
steadyStates(mc)
```

```
AA Aa aa
1 0 0
```

Let's assume that initially the gene *a* is present in the 1% of the population in the combination *aa*, that is, $p_{AA} = 0.99$, $p_{Aa} = 0$, and $p_{aa} = 0.01$. We run the R code below to see how the population genetic composition changes from generation to generation by computing recursively $p_{n+1} = p_n \cdot \mathbf{P}$, $n = 1, 2, 3$, etc., with the initial condition $p_1 = (0.99, 0, 0.01)$.

```

library(expm)
gen1<- c(0.99, 0, 0.01)
gen<- gen1%*%tm
for (n in 2:10) {
  print(n)
  print(round(gen, digits=10))
  gen<- gen%*%tm
}

```

n	p_{AA}	p_{Aa}	p_{aa}
1	0.99	0	0.01
2	0.99	0.01	0
3	0.995	0.005	0
4	0.9975	0.0025	0
5	0.99875	0.00125	0
6	0.999375	0.000625	0
7	0.9996875	0.0003125	0
8	0.9998438	0.00015625	0
9	0.999921875	0.000078125	0
10	0.9999609375	0.0000390625	0

Note that already in the second generation the gene type aa disappears, and is transformed into the hybrid type Aa , and that after as many as ten generations, the gene a is still lingering on in the population. \square

APPLICATION 1.4. In this application, we present yet another chain that is not Markov. It has to do with weather conditions. On an intuitive level, weather tomorrow depends not just on today's weather but on the weather for several past days, if not the entire history of weather conditions in the region. To support this intuitive supposition numerically, we downloaded from *kaggle.com* an open-access historical hourly weather data for 2012-2017 (file "weather_description.csv"), focused only on the column for Los Angeles, and clumped the weather conditions into the four categories

$$S = \{s = \text{"sky clear"}, c = \text{"cloudy"}, f = \text{"fog"}, r = \text{"rain"}\}.$$

We then found empirically the conditional probability of clear sky tomorrow, given clear skies yesterday and today,

$$\hat{\mathbb{P}}(X_3 = s \mid X_2 = s, X_1 = s) = \frac{\hat{\mathbb{P}}(sss)}{\hat{\mathbb{P}}(sss) + \hat{\mathbb{P}}(ssc) + \hat{\mathbb{P}}(ssf) + \hat{\mathbb{P}}(ssr)} = 0.9647,$$

and the conditional probability of clear sky tomorrow, given clear sky today,

$$\hat{\mathbb{P}}(X_3 = s \mid X_2 = s) = \frac{\hat{\mathbb{P}}(sss) + \hat{\mathbb{P}}(css) + \hat{\mathbb{P}}(fss) + \hat{\mathbb{P}}(rss)}{\hat{\mathbb{P}}(sss) + \hat{\mathbb{P}}(css) + \cdots + \hat{\mathbb{P}}(rsr)} = 0.9356.$$

Since these two estimates are not the same, we concluded that hourly weather conditions don't form a Markov chain. The complete R code and output follow.

```
weather.data<- read.csv("./weather_description.csv",
header=TRUE, sep=",")

LA<- weather.data$Los.Angeles

X3<- ifelse(LA=="sky is clear", "clear", ifelse(LA %in%
c("broken clouds", "few clouds", "overcast clouds",
"scattered clouds"), "clouds", ifelse(LA %in% c("light
intensity drizzle", "dust", "fog", "haze", "mist", "smoke",
"drizzle"), "fog", "rain")))

library(Hmisc) #Harrell Miscellaneous packages
X2<- Lag(X3,shift=1)
X1<- Lag(X3,shift=2)

sss<- ifelse(X1=="clear"& X2=="clear"& X3=="clear",1,0)
css<- ifelse(X1=="clouds"& X2=="clear"& X3=="clear",1,0)
fss<- ifelse(X1=="fog"& X2=="clear"& X3=="clear",1,0)
rss<- ifelse(X1=="rain"& X2=="clear"& X3=="clear",1,0)
ssc<- ifelse(X1=="clear"& X2=="clear"& X3=="cloudy",1,0)
csc<- ifelse(X1=="clouds"& X2=="clear"& X3=="clouds",1,0)
fsc<- ifelse(X1=="fog"& X2=="clear"& X3=="clouds",1,0)
rsc<- ifelse(X1=="rain"& X2=="clear"& X3=="clouds",1,0)
ssf<- ifelse(X1=="clear"& X2=="clear"& X3=="fog",1,0)
csf<- ifelse(X1=="clouds"& X2=="clear"& X3=="fog",1,0)
fsf<- ifelse(X1=="fog"& X2=="clear"& X3=="fog",1,0)
rsf<- ifelse(X1=="rain"& X2=="clear"& X3=="fog",1,0)
ssr<- ifelse(X1=="clear"& X2=="clear"& X3=="rain",1,0)
csr<- ifelse(X1=="clouds"& X2=="clear"& X3=="rain",1,0)
fsr<- ifelse(X1=="fog"& X2=="clear"& X3=="rain",1,0)
rsr<- ifelse(X1=="rain"& X2=="clear"& X3=="rain",1,0)

#computing P(X3=s|X2=s,X1=s)
sum(sss)/sum(sss+ssc+ssf+ssr)
```

0.9647471

```
#computing P(X3=s|X2=s)
sum(sss+css+fss+rss)/sum(sss+css+fss+rss+ssc+csc+fsc+rsc+ssf
+csf+fsf+rsf+ssr+csr+fsr+rsr)
```

0.9355981

□

Exercises

EXERCISE 1.1. A Markov chain has a one-step transition probability matrix

$$\begin{array}{c} \begin{array}{ccc} & 1 & 2 & 3 \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \\ 0.8 & 0.1 & 0.1 \end{bmatrix} \end{array}.$$

Compute the following probabilities:

- (a) $\mathbb{P}(X_3 = 2 \mid X_0 = 1, X_1 = 2, X_2 = 3)$.
- (b) $\mathbb{P}(X_4 = 3 \mid X_0 = 2, X_3 = 1)$.
- (c) $\mathbb{P}(X_0 = 1, X_1 = 2, X_2 = 3, X_3 = 1)$. Assume $\mathbb{P}(X_0 = 1) = 1$.
- (d) $\mathbb{P}(X_0 = 1, X_1 = 2, X_3 = 3, X_5 = 1)$. Assume $\mathbb{P}(X_0 = 1) = 1$.

EXERCISE 1.2. Consider a Markov chain with the transition probability matrix

$$\begin{array}{c} \begin{array}{ccccc} & 1 & 2 & 3 & 4 & 5 \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.5 & 0.0 & 0.0 & 0.0 & 0.5 \\ 0.2 & 0.0 & 0.0 & 0.0 & 0.8 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \end{bmatrix} \end{array}.$$

- (a) Plot a diagram of the Markov chain.
- (b) Identify all transient and recurrent classes. Identify all absorbing and reflective states. Find the period of each state.
- (c) Simulate three trajectories of the chain that start at a randomly chosen state. Comment on what you see.
- (d) Find the steady-state probabilities and interpret them. Is it an ergodic chain?
- (e) Plot the unconditional probabilities at time n against the time and comment on how fast the probabilities converge to the steady-state distribution.

EXERCISE 1.3. Consider a Markov chain with the one-step transition probability matrix

$$\begin{array}{c}
 \begin{array}{ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
 \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} & \left[\begin{array}{ccccccc}
 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0.4 & 0.2 & 0.2 & 0.2 \\
 0 & 0 & 0 & 0 & 0.2 & 0.4 & 0.4 \\
 0.3 & 0 & 0 & 0.1 & 0.3 & 0.1 & 0.2 \\
 0 & 0 & 0 & 0.2 & 0.2 & 0.3 & 0.3 \\
 0 & 0 & 0 & 0.5 & 0.2 & 0.2 & 0.1
 \end{array} \right]
 \end{array}
 \end{array}$$

- Plot a diagram of the Markov chain.
- Identify all recurrent and transient classes. Find their periods. Are there any absorbing and reflecting states?
- Simulate two trajectories of the chain that start at a randomly selected state. Discuss what you see in the plot.
- Calculate the limiting probabilities and interpret them. Is the chain ergodic?
- Plot the unconditional probability vectors p_n against n and comment on the speed of convergence to the limiting distribution.

EXERCISE 1.4. Consider a Markov chain with the one-step transition probability matrix

$$\begin{array}{c}
 \begin{array}{ccccc}
 & 1 & 2 & 3 & 4 & 5 \\
 \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \left[\begin{array}{ccccc}
 0.1 & 0.2 & 0.3 & 0 & 0.4 \\
 0 & 0.5 & 0.5 & 0 & 0 \\
 0 & 1 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 1 \\
 0 & 0 & 0 & 0.6 & 0.4
 \end{array} \right]
 \end{array}
 \end{array}$$

- Plot the diagram of the Markov chain.
- Find all recurrent and transient classes and their periods. Are there any absorbing or reflecting states?
- Simulate several trajectories of the Markov chain and discuss the patterns that you see.
- Show that the chain is non-ergodic because there are two invariant probability measures. Which one of them is the stationary distribution?
- Plot the graphs of unconditional probabilities against time, assuming successively that the chain starts in states 1, 2, 3, 4, and 5. Interpret each graph.

EXERCISE 1.5. In a box there are two red (R), four blue (B), and eight green (G) balls. One ball is drawn at a time and its color is noted. Consider the stochastic process $\{X_n, n = 1, 2, \dots\}$ with the state space $S = \{R, B, G\}$.

(a) Show that this process is not a Markov chain, if the drawing is done without replacement. Hint: Show, for instance, that $\mathbb{P}(X_3 = G | X_1 = R, X_2 = B) \neq \mathbb{P}(X_3 = G | X_1 = G, X_2 = B)$.

(b) Show that this process is a Markov chain, if the drawing is done with replacement.

EXERCISE 1.6. Consider a sequence of heads and tails obtained by a series of independent flips of a fair coin. Show that it can be modeled by a Markov chain with the state space $S = \{H, T\}$. Find the transition probability matrix and the limiting distribution.

EXERCISE 1.7. Assume that the usage of vowels and consonants in “Moby Dick” by Herman Melville can be modeled by a Markov chain.

(a) Find the transition probability matrix for [Chapter 1](#) of this novel. Calculate the limiting probabilities and verify that they are equal to the overall proportions of vowels and consonants in the text.

(b) Do states in [Chapter 2](#) follow the same transition probability matrix as those in [Chapter 1](#)?

EXERCISE 1.8. A student at a secretarial school typed the sentence “The quick brown fox jumped over the lazy dog” 500 times. Show that the resulting text cannot be modeled as a Markov chain.

EXERCISE 1.9. Consider the Mendelian gene inheritance model introduced in Application 1.3. Suppose that the second parent is chosen randomly from the gene pool with all types AA , Aa , or aa .

(a) Show that the genotype of the offspring follows a Markov chain with the state space $S = \{(AA, AA), (AA, Aa), (AA, aa), (Aa, AA), (Aa, Aa), (Aa, aa), (aa, AA), (aa, Aa), (aa, aa)\}$ and the transition probability matrix

$$\begin{array}{c}
 \begin{array}{c} (AA, AA) \\ (AA, Aa) \\ (AA, aa) \\ (Aa, AA) \\ (Aa, Aa) \\ (Aa, aa) \\ (aa, AA) \\ (aa, Aa) \\ (aa, aa) \end{array}
 \begin{bmatrix}
 (AA, AA) & (AA, Aa) & (AA, aa) & (Aa, AA) & (Aa, Aa) & (Aa, aa) & (aa, AA) & (aa, Aa) & (aa, aa) \\
 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\
 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 0 & 0 \\
 1/12 & 1/12 & 1/12 & 1/6 & 1/6 & 1/6 & 1/12 & 1/12 & 1/12 \\
 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\
 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\
 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3
 \end{bmatrix}
 \end{array}$$

(b) Determine the transient and recurrent classes of the Markov chain.

(c) Find the stationary distribution. What is the steady-state genetic composition for both parents?

(d) Which initial state achieves the stationary distribution in the smallest number of generations? Which in the largest? Assume the precision of four correct decimals after rounding.

EXERCISE 1.10. Refer to Application 1.4. Consider the data in the file “weather_description.csv.” Choose a city other than Los Angeles and conduct the analysis similar to the one given the application.

EXERCISE 1.11. On an intuitive level, pollution level for a region doesn’t follow a Markov chain as the pollution level tomorrow depends on pre-history and not just on today’s level. But suppose that for some areas, air quality status (good/unhealthy/hazardous) depends only on those in the previous two days, and not in earlier days. Show that in this case, we can look at two days at a time and model data as a Markov chain.

EXERCISE 1.12. Consider a simplified monopoly game with only five squares and respective incomes of \$200, \$0, -\$75, \$105, and -\$130. A player starts at the first square, rolls a fair die once, and moves forward as many steps as the die shows.

- Argue that this game can be modeled as a Markov chain and find its transition probability matrix.
- Compute the steady-state probability of each square, and find the long-run winning of the player.

EXERCISE 1.13. Suppose that road traffic conditions can be modeled as a Markov chain with the state space $S = \{\text{light}, \text{heavy}, \text{jammed}\}$, and suppose that traffic conditions change every 20 minutes. Assume that between 1PM

and 4PM, the transition probability matrix is $\begin{bmatrix} 0.4 & 0.4 & 0.2 \\ 0.3 & 0.5 & 0.2 \\ 0 & 0.5 & 0.5 \end{bmatrix}$, whereas between 4PM and 6PM it changes to $\begin{bmatrix} 0.1 & 0.5 & 0.4 \\ 0.1 & 0.3 & 0.6 \\ 0 & 0.1 & 0.9 \end{bmatrix}$.

- If the traffic starts with the **light** state at 1PM, what is the distribution of the states at 6PM?
- Simulate 10,000 trajectories to verify the result of part (a).

EXERCISE 1.14. A certain species of shrubs has four states: state 1 if it is sustainable, state 2 if it is threatened, state 3 if it is endangered, and state 4 if it is extinct. Plant assessment surveys are done at regular time intervals.

Transitions between states are modeled by a Markov chain with the transition probability matrix

$$\begin{array}{c} \begin{array}{cccc} & 1 & 2 & 3 & 4 \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0.6 & 0.2 & 0.1 & 0.1 \\ 0.7 & 0.2 & 0.1 & 0.0 \\ 0.1 & 0.3 & 0.4 & 0.2 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix} \end{array} \end{array}.$$

- (a) Assuming that a shrub is initially sustainable, simulate several trajectories of the Markov chain.
- (b) Find the probability that initially sustainable shrub will eventually become extinct.

EXERCISE 1.15. A music instrument store is open every day of the week except Monday. During that day, if the inventory count is below 3, more instruments are ordered, so that by Tuesday morning there are 7 instruments in stock. If 3 or more instruments are in stock, then no action is taken. The number of instruments sold during the business days is a Poisson random variable with a mean of 4. Any demand that cannot be satisfied is lost. (a) Argue that the inventory each Tuesday morning can be modeled as a Markov chain. Find its state space and the one-step transition probability matrix.

(b) Generate inventory trajectories, assuming that the initial inventory size is randomly chosen.

(c) Suppose one week there are 7 instruments in stock on Tuesday morning. Compute the probability that there will be 7 instruments in stock also on each of the three subsequent Tuesday mornings.

(d) The weekly storage cost is \$5 per instrument that is in the store on Tuesday morning. Compute the long-run expected weekly storage cost.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

2

Random Walk

2.1 Definition of Random Walk

Consider a random process which state space comprises all integers on a real line. The process starts at zero and transitions either one step to the right with probability p or one step to the left with probability $1 - p$. This process is called a *simple random walk*. It is called *symmetric* if $p = 1/2$ and *asymmetric*, otherwise. It is also termed *infinite* because it is defined on an infinite set of integers. If a random walk occurs on a finite subset of integers, it is termed a *finite-state random walk* or a *random walk on a finite grid* or a *bordered random walk*.

In this chapter, we also consider some variations of a random walk, such as two-, three-, and higher-dimensional random walks, and random walks on graphs. Note that in general, random walks don't have to start at the origin. They can start at any randomly chosen starting point. Below we give formal definitions of all these processes.

A *simple (asymmetric, one-dimensional, infinite) random walk*¹ is a special case of a Markov chain which state space consists of integers $S = \{0, \pm 1, \pm 2, \dots\}$, and the transition probabilities are of the form $p_{i,i+1} = \mathbb{P}(X_{n+1} = i + 1 | X_n = i) = p$ and $p_{i,i-1} = \mathbb{P}(X_{n+1} = i - 1 | X_n = i) = 1 - p$, $i = 0, \pm 1, \pm 2, \dots$. The transition probability matrix for a random walk is

$$\begin{array}{cccccc} & \dots & -2 & -1 & 0 & 1 & 2 & \dots \\ \dots & \left[\begin{array}{cccccc} & & & \dots & \dots & \dots & \\ -2 & \dots & 0 & p & 0 & 0 & 0 & \dots \\ -1 & \dots & 1-p & 0 & p & 0 & 0 & \dots \\ 0 & \dots & 0 & 1-p & 0 & p & 0 & \dots \\ 1 & \dots & 0 & 0 & 1-p & 0 & p & \dots \\ 2 & \dots & 0 & 0 & & 1-p & 0 & \dots \\ \dots & & & \dots & \dots & \dots & & \end{array} \right] \end{array}.$$

A *symmetric random walk in two dimensions* is defined as a random walk on an integer lattice that moves right or left or up or down with a probability

¹The term was first used in Pearson K. (1905). "The Problem of the Random Walk". *Nature*, 72: 294.

1/4. Likewise, a *symmetric random walk in d dimensions* is a random walk on the d -dimensional integer lattice, with equiprobable moves in $2d$ directions (with probability $1/(2d)$).

Some variations of a simple random walk include a *random walk with loops (or delays)*, in the sense that the allowed moves are to the right with probability p , to the left with probability q , and remaining in the same state with probability $1 - p - q$. Also, in two dimensions, it might be allowed to move diagonally as well as to stay in the same place, so each move has a probability of $1/9$. Or a *finite-state random walk* might be defined on a finite integer grid with the boundary (or border, or barrier) states being either reflecting or absorbing. In addition, it is possible to define a *random walk on a graph*, where at every state, the process chooses among all neighboring states with equal probability. Below we consider some examples.

EXAMPLE 2.1. A gambler either wins \$5 with probability 0.55 or loses \$5 with probability 0.45. He starts playing with \$50 and plays until he either goes broke or doubles his original amount. This is an example of a finite one-dimensional random walk with absorbing barriers, since, once entered, the states \$0 and \$100 are never left. The transition probability matrix is

	\$0	\$5	\$10	...	\$45	\$50	\$55	...	\$95	\$100
\$0	1	0	0	...	0	0	0	...	0	0
\$5	0.45	0	0.55	...	0	0	0	...	0	0
\$10	0	0.45	0	...	0	0	0	...	0	0
...	
\$45	0	0	0	...	0	0.55	0	...	0	0
\$50	0	0	0	...	0.45	0	0.55	...	0	0
\$55	0	0	0	...	0	0.45	0	...	0	0
...	
\$95	0	0	0	...	0	0	0	...	0	0.55
\$100	0	0	0	...	0	0	0	...	0	1

If, for example, a gambler who goes broke can take a credit that brings him back to the \$50 fortune, the state \$0 is no longer an absorbing state, but is, in fact, a reflecting state. On the other end of his wealth spectrum, once the gambler reaches the \$100 fortune, he returns the credit (with, say, 10% interest) and is bounced to \$45, and thus the \$100 state is reflecting as well. \square

EXAMPLE 2.2. Suppose that a mouse is running through a maze with rooms A, B, C, D , and E , and two exits (see the illustration by Shayan Khatri). The mouse runs along the passages, and in any given room, it decides randomly which passage to take.

PROPOSITION 2.1. (MEAN AND VARIANCE OF A RANDOM WALK). The mean and variance of a random walk at time n are $\mathbb{E}(X_n) = (2p - 1)n$ and $\mathbb{V}ar(X_n) = 4p(1 - p)n$.

PROOF: We can write $X_n = Z_1 + \dots + Z_n$ where Z_i 's are independent random variables with the binary distribution:

$$Z_i = \begin{cases} 1, & \text{with probability } p, \\ -1, & \text{with probability } 1 - p, \end{cases} \quad i = 1, \dots, n.$$

Note that $\mathbb{E}(Z_i) = (1)(p) + (-1)(1 - p) = 2p - 1$, and $\mathbb{V}ar(Z_n) = (1)^2(p) + (-1)^2(1 - p) - (2p - 1)^2 = 1 - (2p - 1)^2 = 4p - 4p^2 = 4p(1 - p)$. Thus, we compute $\mathbb{E}(X_n) = \mathbb{E}(Z_1) + \dots + \mathbb{E}(Z_n) = (2p - 1)n$, and $\mathbb{V}ar(X_n) = \mathbb{V}ar(Z_1) + \dots + \mathbb{V}ar(Z_n) = 4p(1 - p)n$. \square

Note that for a symmetric random walk, $p = 1/2$, and hence, $\mathbb{E}(X_n) = 0$ and $\mathbb{V}ar(X_n) = n$, and after n steps, a typical distance from the origin is on the order of \sqrt{n} .

PROPOSITION 2.2. (PROBABILITY OF A RETURN). The probability that a one-dimensional random walk returns to the starting point in exactly n steps is $\binom{n}{n/2} p^{n/2} (1 - p)^{n/2}$ if n is even, and zero, otherwise.

PROOF: First of all, note that the random walk can return to the starting point only in an even number of steps, half of which it should be moving to the right (with probability p), and half, to the left (with probability $1 - p$).

There are $\binom{n}{n/2}$ paths with this property, hence:

$$\mathbb{P}(X_n = 0 \mid X_0 = 0) = \begin{cases} \binom{n}{n/2} p^{n/2} (1 - p)^{n/2}, & \text{if } n \text{ is even,} \\ 0, & \text{if } n \text{ is odd} \end{cases}.$$

Note that the random walk can cross the starting point several times before ending there in exactly n steps. \square

The proofs of the next two propositions are omitted as they lie beyond the scope of this book.

PROPOSITION 2.3. (RECURRENCE VS. TRANSIENCE OF 1D RANDOM WALK). A one-dimensional random walk will come back to the origin infinitely many times with probability one if and only if $p = 1/2$. This means that a symmetric one-dimensional random walk is *recurrent*, while an asymmetric one-dimensional random walk is *transient*. It will, with a positive probability, come back only finite number of times and will eventually wander away.

PROPOSITION 2.4. (RECURRENCE VS. TRANSIENCE OF d -DIM RANDOM WALK). A two-dimensional random walk is recurrent if and only if it is symmetric (i.e., it goes up or down or left or right with probability $1/4$). Any random walk (symmetric or not) in 3D or a higher dimension is transient. It will eventually wander away from the origin with a positive probability.

Thence, a bug crawling randomly along a railroad track will visit every inch of it infinitely many times. Likewise, a King moving randomly on an infinite chessboard will visit every square infinitely many times, whereas, say, a drone performing a random walk in the sky will eventually go into outer space, never to be seen again.

2.3 Simulations in R

SIMULATION 2.1. Below we simulate three trajectories of a one-dimensional random walk. We assume the walk starts at zero, and specify p as 0.6 and the number of steps as 25.

```
#specifying parameters
ntraj<- 3
p<- 0.6
nsteps<- 25

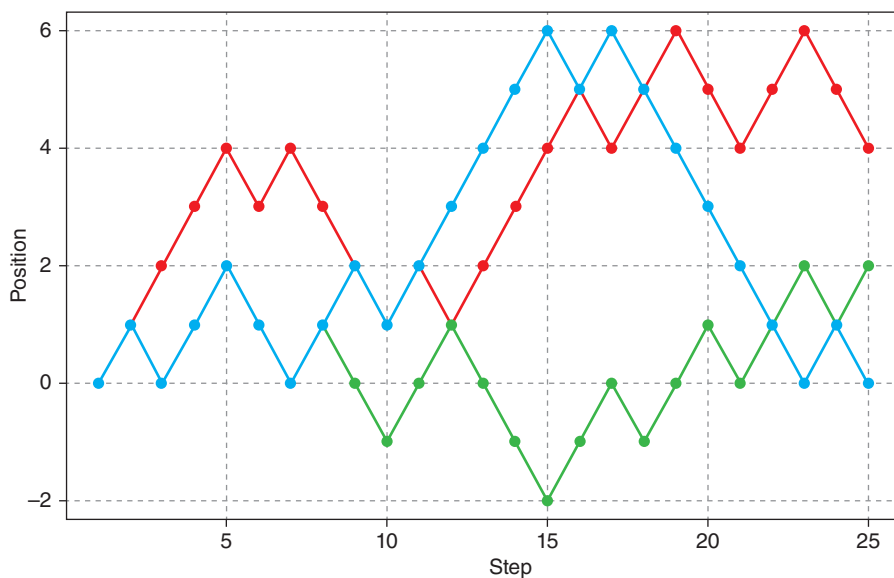
#specifying seed
set.seed(45568223)

#defining walk as matrix
walk<- matrix(NA, nrow=nsteps, ncol=ntraj)

#simulating trajectories
for (j in 1:ntraj) {
  walk[1,j]<- 0
  for (i in 2:nsteps)
    walk[i,j]<- ifelse(runif(1)<p, walk[i-1,j]+1, walk[i-1,j]-1)
}

#plotting trajectories
matplot(walk, type="l", lty=1, lwd=2, col=2:4,
  ylim=c(range(walk)), xlab="Step", ylab="Position",
  panel.first=grid())
```

```
points(1:nsteps, walk[,1], pch=16, col=2)
points(1:nsteps, walk[,2], pch=16, col=3)
points(1:nsteps, walk[,3], pch=16, col=4)
```



□

SIMULATION 2.2. The code below simulates and plots a two-dimensional random walk with a total of 10,000 steps, emanating from the origin.

```
#specifying number of steps
nsteps<- 10000

#specifying seed
set.seed(607335)

#defining walk as matrix
walk<- matrix(NA, nrow=nsteps, ncol=2)

#setting starting point
walk[1,]<- c(0,0)

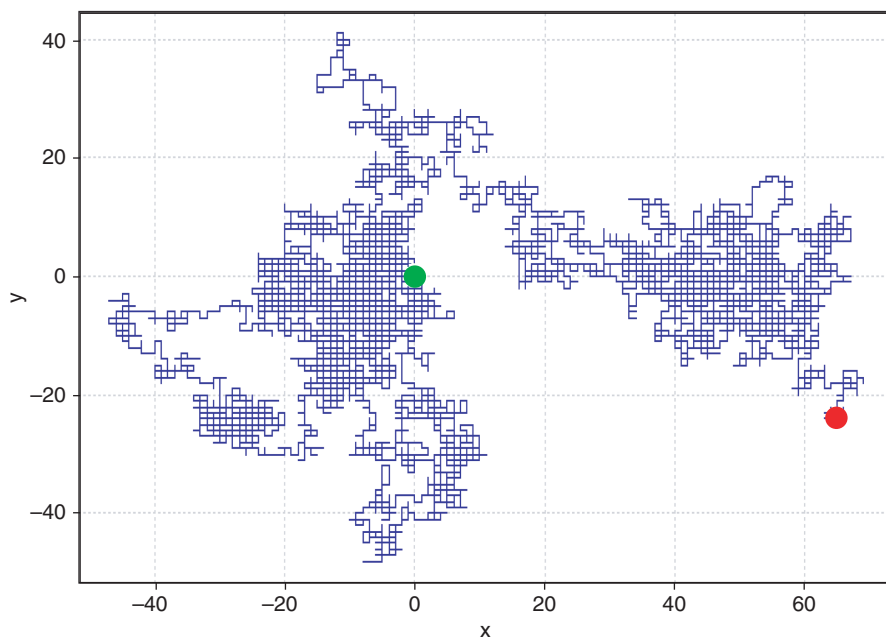
#defining random steps
rstep<- matrix(c(1, 0, -1, 0, 0, 1, 0, -1), nrow=4, ncol=2,
byrow=TRUE)
```

```
#simulating trajectories
for (i in 2:nsteps)
  walk[i,]<- walk[i-1,] + rstep[sample(1:4, size=1),]

#plotting trajectories
plot(x=walk[,1], y=walk[,2], type="l", col="blue",
     xlim=range(walk[,1]), ylim=range(walk[,2]), xlab="x",
     ylab="y", panel.first=grid())

#adding starting point
points(cbind(walk[1,1], walk[1,2]), pch=16, col="green",
       cex=2)

#adding ending point
points(cbind(walk[nsteps,1], walk[nsteps,2]), pch=16,
       col="red", cex=2)
```



□

SIMULATION 2.3. The following code simulates and plots a three-dimensional random walk with 5,000 steps.

```
#specifying number of steps
nsteps<- 5000

#specifying seed
set.seed(830126)

#defining walk as matrix
walk<- matrix(NA, nrow=nsteps, ncol=3)

#setting starting point
walk[1,]<- c(0,0,0)

#defining random steps
rstep<- matrix(c(1,0,0,-1,0,0,0,1,0,0,-1,0,0,1,0,0,-1),
nrow=6, ncol=3, byrow=TRUE)

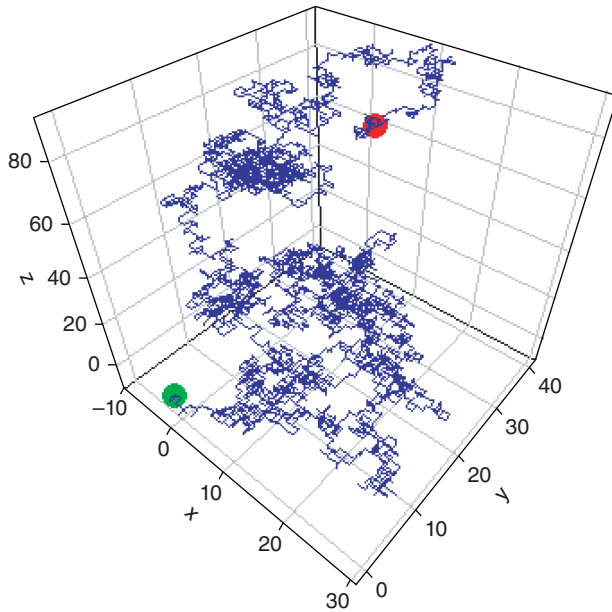
#simulating trajectories
for (i in 2:nsteps)
walk[i,]<- walk[i-1,]+rstep[sample(1:6, size=1),]

#plotting trajectories
library(plot3D)

lines3D(walk[,1], walk[,2], walk[,3], col="blue",
xlim=range(walk[,1]), ylim=range(walk[,2]),
zlim=range(walk[,3]), xlab="x", ylab="y", zlab="z", bty="b2",
ticktype="detailed")

#adding starting point
points3D(x=walk[1,1], y=walk[1,2], z=walk[1,3], add=TRUE,
pch=16, col="green", cex=2)

#adding ending point
points3D(walk[nsteps,1], walk[nsteps,2], walk[nsteps,3],
add=TRUE, pch=16, col="red", cex=2)
```



□

2.4 Applications of Random Walk

APPLICATION 2.1. (GAMBER'S RUIN PROBLEM). Gambling is perhaps the oldest area of application of random walks. And the most famous is the *Gambler's Ruin Problem*. A version of the gambler's ruin problem has been formulated as early as 1656, in correspondence between Blaise Pascal and Pierre de Fermat.

Suppose a gambler starts with a fortune of $\$i$ and will move up $\$1$ with probability p or down $\$1$ with probability $q = 1 - p$ until he is either broke or reaches the fortune of $\$N$. What is the probability that he goes broke?

To answer this question, we will compute the complementary probability of reaching the fortune of $\$N$. Denote by \mathbf{P}_j the probability of winning $\$N$ if a gambler starts with $\$j$. Conditioning on the outcome of the first move, we can write the recurrence relation: $\mathbf{P}_j = q\mathbf{P}_{j-1} + p\mathbf{P}_{j+1}$ with the boundary conditions $\mathbf{P}_0 = 0$ and $\mathbf{P}_N = 1$. We can write this relation as $p\mathbf{P}_j + q\mathbf{P}_j = q\mathbf{P}_{j-1} + p\mathbf{P}_{j+1}$ and rewrite as $\mathbf{P}_{j+1} - \mathbf{P}_j = \frac{q}{p}(\mathbf{P}_j - \mathbf{P}_{j-1})$. From

here, we see that $\mathbf{P}_2 - \mathbf{P}_1 = \frac{q}{p}(\mathbf{P}_1 - \mathbf{P}_0) = \frac{q}{p}\mathbf{P}_1$, $\mathbf{P}_3 - \mathbf{P}_2 = \frac{q}{p}(\mathbf{P}_2 - \mathbf{P}_1) = \left(\frac{q}{p}\right)^2 \mathbf{P}_1, \dots, \mathbf{P}_i - \mathbf{P}_{i-1} = \left(\frac{q}{p}\right)^{i-1} \mathbf{P}_1$. Summing up the identities, we obtain $\mathbf{P}_i - \mathbf{P}_1 = \left[\frac{q}{p} + \left(\frac{q}{p}\right)^2 + \dots + \left(\frac{q}{p}\right)^{i-1}\right] \mathbf{P}_1$. From here,

$$\mathbf{P}_i = \begin{cases} \frac{1 - (q/p)^i}{1 - q/p} \mathbf{P}_1, & \text{if } q/p \neq 1, \\ i\mathbf{P}_1, & \text{if } q/p = 1. \end{cases}$$

To find \mathbf{P}_1 , we use the boundary condition $\mathbf{P}_N = 1$. It yields

$$\mathbf{P}_1 = \begin{cases} \frac{1 - q/p}{1 - (q/p)^N}, & \text{if } q/p \neq 1, \\ 1/N, & \text{if } q/p = 1. \end{cases}$$

Consequently, if gambling is modeled as a symmetric random walk (with $p = q = 1/2$), the probability of reaching the fortune $\$N$ is i/N , and thus, the probability of ruin is $(N - i)/N$. If the model is an asymmetric random walk, the probability of reaching $\$N$ is $\mathbf{P}_i = \frac{1 - (q/p)^i}{1 - (q/p)^N}$, and hence, the probability of ruin is $1 - \mathbf{P}_i = \frac{(q/p)^i - (q/p)^N}{1 - (q/p)^N}$.

Also, it can be shown (see Exercise 2.7) that the expected number of games that the gambler plays until he reaches $\$N$ or goes bankrupt is

$$\mathbf{E}_i = \begin{cases} \frac{N}{q-p} \frac{(q/p)^i - (q/p)^N}{1 - (q/p)^N} - \frac{N-i}{q-p} = \frac{i - N\mathbf{P}_i}{q-p}, & \text{if } q/p \neq 1, \\ i(N - i), & \text{if } q/p = 1. \end{cases}$$

Let us see how it plays out with some specific values. In Example 2.1, the gambler started with $\$50$, goes up $\$5$ with probability 0.55, or goes down $\$5$ with probability 0.45. He would end up with either $\$100$ or $\$0$. Since the increment is $\$5$, in our notation this translates into $i = 10$, $N = 20$, and $p = 0.55$. The probability of reaching $\$100$ is $\frac{1 - (0.45/0.55)^{10}}{1 - (0.45/0.55)^{20}} = 0.8815$ and the probability of ruin is, respectively, $1 - 0.8815 = 0.1185$.

As for the expected number of games, we compute $\mathbf{E}_i = \frac{10 - (20)(0.8815)}{0.45 - 0.55} = 76.3$. That is, the gambler plays, on average, 76.3 games.

We can verify these probabilities and the expected number of games empirically, by running the following R code that simulates 100,000 trajectories and counts how many of them ended in 20, how many ended in 0, and keeps track of the total number of games played until the end is reached. These values are then averaged over the total number of trajectories.

```
#specifying parameters
p<- 0.55
i<- 10
N<- 20
ntraj<- 100000

#defining walk as vector
walk<- c()

#setting counters
nNs<- 0
nzeros<- 0
ngames<- 0

#setting seed number
set.seed(30112443)

#simulating trajectories until hitting N or 0
for (j in 1:ntraj) {
  walk[1]<- i
  k<- 2
  repeat {
    walk[k]<- ifelse(runif(1)<p, walk[k-1]+1, walk[k-1]-1)
    ngames<- ngames + 1

    if (walk[k]==N) {
      nNs<- nNs+1
      break
    }

    else if(walk[k]==0) {
      nzeros<- nzeros+1
      break
    }

    k<- k+1
  }
}

print(prob.Ns<- nNs/ntraj)
```


0.88279

```
print(prob.zeros <- nzeros/ntraj)
```

0.11721

```
print(mean.ngames<- ngames/ntraj)
```

76.18866

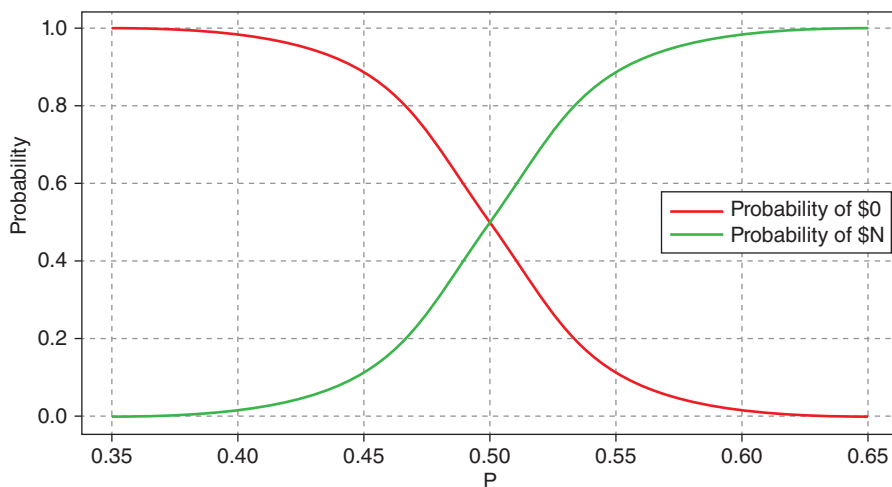
Next, we can plot the graph of the probabilities as a function of p , for our specific values of $i = 10$ and $N = 20$. The syntax and the graph follow. The green curve depicts the probability of reaching N , whereas the red one displays the probability of ruin.

```
p<- seq(0.35,0.65,0.001)
i<- 10
N<- 20
q<- 1-p
p.ruin<- ifelse(p==0.5, (N-i)/N, ((q/p)^i-(q/p)^N)/(1-(q/p)
^N))

#ploting the graphs
plot(p, p.ruin, type = "l", lwd=2, col = "red", xlab="p",
ylab="Probability", panel.first = grid())

lines(p, 1-p.ruin, lwd=2, col = "green")

legend("right", c("Probability of $0", "Probability of $N"),
lty=1, col=2:3)
```

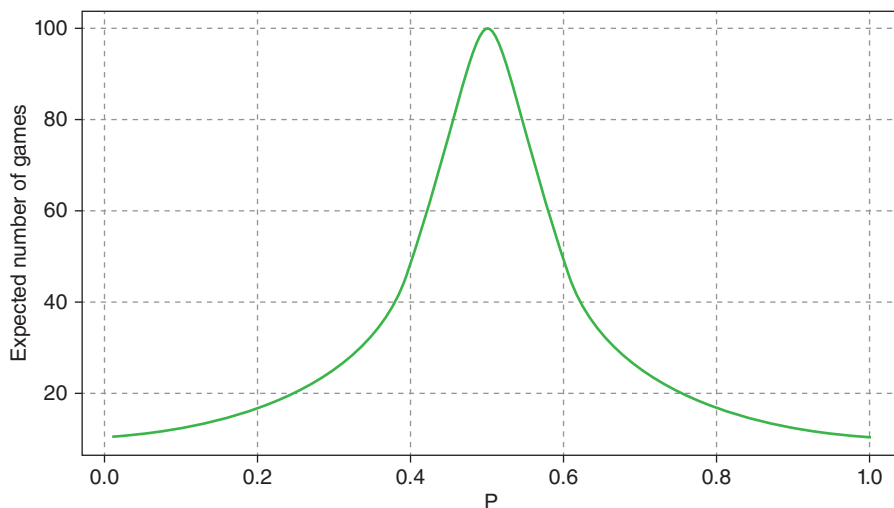


Note from the graph that the probability p doesn't have to be very small for the ruin to happen almost certainly. For p ranging between 0.4 and 0.6, the probability of ruin goes down from almost 1 to almost 0. Respectively, the probability of reaching N increases from almost 0 to almost 1.

Finally, we plot the graph of the expected number of games against p . The code and the output are presented below.

```
p<- seq(0.01,1,0.01)
i<- 10
N<- 20
q<- 1-p
E.ngames<-
ifelse(p==0.5,i*(N-i),(i-N*(1-(q/p)^i)/(1-(q/p)^N))/(1-2*p))

plot(p, E.ngames, type="l", lwd=2, col="green", xlab="p",
ylab="Expected number of games", panel.first=grid())
```



Note that the maximum of this function is achieved at $p = 1/2$ and the maximum value is $i(N - i) = (10)(20 - 10) = 100$. Also, the graph is symmetric due to the starting point being right in the middle, i.e., $i = N/2$. \square

APPLICATION 2.2. (RANDOM WALK ON A GRAPH). Consider the random walk through the maze introduced in Example 2.2. Suppose the mouse starts in room C and spends one second on each transition from room to room (or *Exit*). We want to calculate how many seconds, on average, the mouse spends in the maze before exiting. We argue as follows. The mouse will spend exactly k seconds in the maze if it exits in exactly k transitions. So, we need to find the probability to transition from room C to an *Exit* in exactly k steps. Let \mathbf{P} denote the one-step transition probability matrix. Then

$$(0, 0, 0, 1, 0, 0) \mathbf{P}^k (1, 0, 0, 0, 0, 0)^{-1}$$

gives the probability to transition from vertex C to *Exit* in k or fewer steps, and thus,

$$(0, 0, 0, 1, 0, 0) \left(\mathbf{P}^k - \mathbf{P}^{k-1} \right) (1, 0, 0, 0, 0, 0)^{-1}$$

is the probability to transition from C to *Exit* in exactly k steps. Consequently, the formula for the expected time that the mouse spends in the maze before it reaches an *Exit* is

$$\begin{aligned} \mathbb{E}(\text{time to exit}) = & (0, 0, 0, 1, 0, 0) \left((1)\mathbf{P} + (2)(\mathbf{P}^2 - \mathbf{P}) \right. \\ & \left. + (3)(\mathbf{P}^3 - \mathbf{P}^2) + (4)(\mathbf{P}^4 - \mathbf{P}^3) + \dots \right) (1, 0, 0, 0, 0, 0)^{-1}. \end{aligned}$$

We run an R code to estimate this expected value. Convergence to the six decimal places that R outputs is achieved with the first 172 terms. Adding more terms doesn't change the output.

```
#specifying transition probability matrix
tm<- matrix(c(1,0,0,0,0,0,1/4,0,1/4,1/4,0,1/4,0,1/2,0,
1/2,0,0,0,1/4,1/4,0,1/4,1/4,1/3,0,0,1/3,0,1/3,0,1/3,0,1/3,1/
3,0), nrow=6, ncol=6, byrow=TRUE)

#setting counter
nsec<- 0

#estimating expected number of seconds
p<- matrix(NA, nrow=172, ncol=6)
p[1,]<- c(0, 0, 0, 1, 0, 0)

for (i in 2:172) {
  p[i,]<- p[i-1,]*%tm
  nsec<- nsec+(i-1)*(p[i,1]-p[i-1,1])
}

print(nsec)
```

9.967213

Hence, the mouse spends, on average, 9.967213 seconds in the maze, making that many transitions between the states (and an *Exit*). \square

Exercises

EXERCISE 2.1. Consider a one-dimensional random walk with the transition probability $p = 0.3$. Simulate 10,000 trajectories of length 50 steps and calculate empirical mean and variance. Are the estimates close to the theoretical values?

EXERCISE 2.2. Consider a symmetric one-dimensional random walk that originates at 0.

(a) Simulate 10,000 trajectories with 1,000 steps each. How many of the trajectories are at point 0 on the 1,000th step?

(b) Find the theoretical probability of returning to 0 on the 1,000th step. Compare to the empirical value.

EXERCISE 2.3. Simulate 10,000 trajectories of 1D, 2D, and 3D symmetric random walks that start at the origin and continue for at most 1,000 steps.

(a) Compute how many of them returned to the origin at least once. Compare the results for different dimensions. Hint: Terminate a trajectory when it returns to the origin.

(b) Consider only the trajectories that returned to the origin within the 1,000 steps. Compute the average number of steps it took those trajectories to return to the origin. Compare the results for different dimensions.

EXERCISE 2.4. Simulate 10,000 trajectories of a two-dimensional symmetric random walk that starts at the origin and continues for a maximum of 1,000 steps.

(a) Estimate the probability of a trajectory ever hitting the vertical barrier $x = 30$.

(b) Estimate the average number of steps it takes a trajectory to hit the barrier, provided it did hit the barrier within the 1,000 steps.

(c) Estimate the expected value of the y -coordinate at the time when the random walk hits the barrier. What should this value be from the theoretical point of view? Hint: deduce from a symmetry argument.

EXERCISE 2.5. Simulate 100 trajectories of a two-dimensional symmetric random walk that starts at the origin and continues for 1,000 steps or until it hits a barrier. The value of the barrier varies between $x = 1$ and $x = 50$. Plot the empirical probability of hitting the barrier against the barrier value. Discuss the pattern you see.

EXERCISE 2.6. Simulate 1,000 trajectories of a two-dimensional symmetric random walk that starts at the origin and continues until it hits a side of a square centered at the origin and having a side length of 20. Estimate the average number of steps that it takes the random walk to reach the square.

EXERCISE 2.7. Suppose a gambler starts with a fortune of $\$i$ and will move up $\$1$ with probability p or down $\$1$ with probability $q = 1 - p$ until he either reaches the fortune of $\$B$ or is down to $\$A$.

(a) Prove that the probability that he reaches $\$B$ before $\$A$ is

$$\mathbf{P}_i = \begin{cases} \frac{(q/p)^A - (q/p)^i}{(q/p)^A - (q/p)^B}, & \text{if } q/p \neq 1, \\ \frac{i-A}{B-A}, & \text{if } q/p = 1. \end{cases}$$

Hint: Show that \mathbf{P}_i solves the recurrence relation $\mathbf{P}_i = p\mathbf{P}_{i+1} + q\mathbf{P}_{i-1}$, with the border constraints $\mathbf{P}_A = 0$ and $\mathbf{P}_B = 1$. Look for the solution in the form $\mathbf{P}_i = c(q/p)^i + d$, if $q/p \neq 1$, and $\mathbf{P}_i = ci + d$ if $q/p = 1$.

(b) Show that the expected number of games the gambler plays until he reaches $\$B$ or $\$A$ is

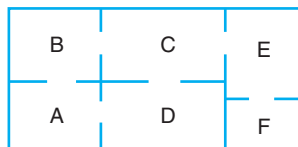
$$\mathbf{E}_i = \begin{cases} \frac{B-A}{q-p} \frac{(q/p)^i - (q/p)^B}{(q/p)^A - (q/p)^B} - \frac{B-i}{q-p}, & \text{if } q/p \neq 1, \\ (B-i)(i-A), & \text{if } q/p = 1. \end{cases}$$

Hint: Show that \mathbf{E}_i satisfies the recurrence relation $\mathbf{E}_i = p\mathbf{E}_{i+1} + q\mathbf{E}_{i-1} + 1$ with the boundary conditions $\mathbf{E}_A = \mathbf{E}_B = 0$. Look for solutions in the form $\mathbf{E}_i = c(q/p)^i + d + i/(q-p)$, if $q/p \neq 1$, and $\mathbf{E}_i = ci + d - i^2$, if $q/p = 1$.

(c) Suppose a gambler comes to a casino with $\$40$ and plays a rigged game with $p = 0.47$ until he doubles the amount or is down to $\$10$ (to pay for a taxi). Calculate the probability that he walks out of the casino with $\$80$. How many games, on average, will he play? Simulate 10,000 trajectories and estimate the probability and the expected length of play.

EXERCISE 2.8. A student visits an Ancient History museum that is open between 9AM and 6PM. He enters the museum at 9AM and wanders the rooms in a random-walk fashion, spending 30 minutes in each room, and then choosing a door at random. The museum floor plan is given in the picture. How

long will the student spend in the museum, on average? Does he expect to leave the museum before it closes for the day? Write down the formula and use R to calculate the result.





Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

3.1 Definition and Must-Know Facts About Poisson Process

A stochastic process $\{N(t), t \geq 0\}$ is called a *counting process* if $N(t)$ gives the total number of events occurring by time t .

A counting process $\{N(t), t \geq 0\}$ is said to have *independent increments* if the number of events that occur in non-overlapping time intervals are independent. For example, $N(5) - N(0)$, the number of events occurring between times 0 and 5, is independent of $N(10) - N(5)$, the number of events occurring between times 5 and 10.

A counting process $\{N(t), t \geq 0\}$ is said to have *stationary increments* if the distribution of the number of events that occur in any time interval depends only on the length of the interval. In other words, $N(t) - N(0)$ and $N(t+s) - N(s)$ have the same distribution that depends only on t and not on s .

A counting process $\{N(t), t \geq 0\}$ is called a *Poisson process*¹ with rate λ , if: (i) no events occur at time 0, i.e., $N(0) = 0$, (ii) it has independent increments, (iii) it has stationary increments, and (iv) $\mathbb{P}(N(t) = n) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$, $n = 0, 1, 2, \dots$. Note that $\mathbb{E}(N(t)) = \text{Var}(N(t)) = \lambda t$.

Since the rate λ of a Poisson process is a constant not depending on time t , the process is sometimes referred to as a *homogeneous* (or *stationary*) Poisson process.

An *interarrival time* is the time between two consecutive occurrences of events. The interarrival time between $(n-1)$ st and n th occurrences will be denoted by T_n , $n = 2, 3, \dots$. The time of the first occurrence will be denoted by T_1 .

¹The reference to a Poisson process first appeared in two independent publications in 1940. The first was the article by Feller, W. (1940). "On the integro-differential equations of purely discontinuous Markov processes." *Trans. Am. Math. Society*, 48(3): 488 – 515. The second was the Ph.D. dissertation by Lundberg, O. (1940). "On random processes and their application to sickness and accident statistics." Uppsala: Almqvist & Wiksell.

The *waiting time* until the n th event (or *event time*), $S_n = T_1 + T_2 + \cdots + T_n$, is the time when the n th event occurs.

PROPOSITION 3.1. Interarrival times T_n , $n = 1, 2, \dots$, are independent exponentially distributed random variables with the density function $f_{T_n}(t) = \lambda e^{-\lambda t}$, $t \geq 0$.

PROOF: Note that $\mathbb{P}(T_1 > t) = \mathbb{P}(N(t) = 0) = e^{-\lambda t}$. Therefore, $T_1 \sim \text{Exp}(\lambda)$. Next, conditioning on the value of the first occurrence, and using independence and stationarity of increments, we write

$$\begin{aligned} \mathbb{P}(T_2 > t) &= \int_0^\infty \mathbb{P}(T_2 > t \mid T_1 = s) \lambda e^{-\lambda s} ds \\ &= \int_0^\infty \mathbb{P}(N(t+s) - N(s) = 0 \mid N(s) = 1) \lambda e^{-\lambda s} ds \\ &= \int_0^\infty \mathbb{P}(N(t) = 0) \lambda e^{-\lambda s} ds = e^{-\lambda t} \int_0^\infty \lambda e^{-\lambda s} ds = e^{-\lambda t}, \end{aligned}$$

that is, $T_2 \sim \text{Exp}(\lambda)$. More generally, conditioning on the time of the n th event occurrence and again using independence and stationarity of increments, we obtain

$$\begin{aligned} \mathbb{P}(T_{n+1} > t) &= \int_0^\infty \mathbb{P}(T_{n+1} > t \mid S_n = s) f_{S_n}(s) ds \\ &= \int_0^\infty \mathbb{P}(N(t+s) - N(s) = 0 \mid N(s) = n) f_{S_n}(s) ds \\ &= \int_0^\infty \mathbb{P}(N(t) = 0) f_{S_n}(s) ds = e^{-\lambda t} \int_0^\infty f_{S_n}(s) ds = e^{-\lambda t}. \quad \square \end{aligned}$$

REMARK 3.1. Recall that exponential is the only continuous distribution that possesses the *memoryless property*, $\mathbb{P}(T > t+s \mid T > t) = \mathbb{P}(T > s)$. In a Poisson process, the increments are independent and stationary and that implies that the process renews itself every moment. Therefore, on an intuitive level, the interarrival times should possess the memoryless property and thus be distributed exponentially. Also, if, say, on average, there are two occurrences per hour ($\lambda = 2$ per hour), the average waiting time between two occurrences is half an hour (mean = $1/\lambda = 1/2$ hour).

PROPOSITION 3.2. The waiting time S_n has *Gamma*(n, λ) distribution. Thus, $\mathbb{E}(S_n) = n/\lambda$ and $\text{Var}(S_n) = n/\lambda^2$. The density function is of the form $f_{S_n}(s) = \frac{\lambda^n s^{n-1}}{(n-1)!} e^{-\lambda s}$, $s \geq 0$.

PROOF: We can write S_n as the sum of interarrival times, i.e., $S_n = T_1 + T_2 + \cdots + T_n$. It is a general fact in the theory of probability that the sum of n independent exponentially distributed random variables with parameter λ has a gamma distribution with parameters n and λ . The quickest proof of this fact is through the moment generating functions. \square

REMARK 3.2. Let's look at a Poisson process as a special case of a Markov chain. The state space of a Poisson process is $S = \{0, 1, 2, \dots\}$. The process jumps from initial state 0 to state 1 with probability 1, then to state 2 with probability 1, etc. The jumps are always of size 1, and transitions between states are allowed only in the direction of increase. Moreover, a Poisson process includes the time component, so it matters how long the process halts between jumps. In fact, we know that it halts an exponentially distributed time. This type of Markov chain is called a *continuous-time Markov chain*.

REMARK 3.3. It is essential to understand that Poisson arrivals occur one at a time. The probability of two simultaneous arrivals is zero since the interarrival time is exponentially distributed and thus the probability of an interarrival time being equal to zero is zero. It means that in situations when, say, people can potentially arrive in groups, one needs to count not arrivals of individual people, but count the arrival of a group as a single event. For example, if we model the process of people joining a ticket line in a movie theater, we would be likely to see an entire party joining the line, so we would count the party as one arrival.

EXAMPLE 3.1. A Poisson process is used to model occurrences of rare events. Here are some instances of Poisson processes: the number of people who enter a store or a bank or a restaurant or a gym or a National Park, the number of cars that pass a certain intersection, the number of auto accidents on a certain stretch of a freeway, the number of births in a hospital, the number of meteors in the night sky, or the number of phone calls to a credit card customer service. Typically, natural disasters occur according to a Poisson process: earthquakes, volcano eruptions, wildfires, etc. Of course, in all the above examples, the considered time period should be short enough for the rate of occurrence to be constant.

Some examples of processes that clearly are not governed by a Poisson law are events that happen according to a schedule, for example, the arrival of buses along a certain route, road closures due to construction work, quarry blasts, building demolitions. Also, events that happen in a competing market, where two rival companies might be scheduling two events at the same time. For instance, two pharmaceutical companies might simultaneously bring

to the market two cardiac medications, or two production companies might release two movies on the same day. In addition, some periodic (or seasonal) natural phenomena cannot be modeled as a Poisson process, i.e., eruptions of Old Faithful geyser in Yellowstone Stone National Park, ocean tides, the appearance of sunspots, etc. \square

EXAMPLE 3.2. Tour buses arrive at a roadside mall with restaurants, bringing 50 tourists each. The times that elapse between consecutive arrivals are independent and exponentially distributed with mean of 15 minutes.

(a) On average, buses arrive every 15 minutes, so, for instance, the expected waiting time for the fifth bus to arrive is $(15)(5) = 75$ minutes, or 1 hour and 15 minutes. Now we put it in our theoretical framework. The rate of arrival of the buses is $\lambda = 4$ per hour. Denote by S_5 the time until the 5th bus arrives. We know that S_5 has a gamma distribution with parameters $n = 5$ and $\lambda = 4$. Therefore, $\mathbb{E}(S_5) = 5/4 = 1.25$ hours, or 1 hour and 15 minutes. We can also compute the variance of S_5 as $\mathbb{V}ar(S_5) = 5/4^2 = 0.3125$ hours squared, and the standard deviation as $\sqrt{\mathbb{V}ar(S_5)} = \sqrt{0.3125} = 0.559$ hours.

(b) The total expected number of tourists who are served lunch at the restaurants between, say, 11AM and 1:30PM is found as follows. Within the two and a half hours, we expect 10 bus arrivals, each carrying 50 tourists. Therefore, we expect a total of $(10)(50) = 500$ tourists. To write it formally, let $N(t)$ denote the number of buses that arrive by time t . We know that $N(t) \sim \text{Poisson}(4t)$. We are given that $t = 2.5$ hours, and therefore, the expected total number of tourists is $(50)\mathbb{E}(N(2.5)) = (50)(4)(2.5) = 500$. \square

EXAMPLE 3.3. Independence and stationarity of increments in a Poisson process allows elegant computations of conditional probabilities. For example, consider a Poisson process with rate $\lambda = 2.2$.

(a) Suppose we want to find the conditional probability that there will be 8 arrivals by time 5 given that there was 1 arrival by time 2. We can argue that since at time 2 the process renews itself, we need to compute the probability that within the next 3 time periods there will be 7 more arrivals. We write $\mathbb{P}(N(5) = 8 | N(2) = 1) = \mathbb{P}(N(5) - N(2) = 7 | N(2) = 1) = \{\text{independence}\} = \mathbb{P}(N(5) - N(2) = 7) = \{\text{stationarity}\} = \mathbb{P}(N(5 - 2) = 7) = \mathbb{P}(N(3) = 7) = \frac{((2.2)(3))^7}{7!} e^{-(2.2)(3)} = 0.147243$.

(b) Similarly, conditional expectations can be computed if we utilize independence and stationarity of increments. $\mathbb{E}[N(9) | N(7) = 10] = \mathbb{E}[N(9) - N(7) | N(7) = 10] + \mathbb{E}[N(7) | N(7) = 10] = \mathbb{E}[N(9) - N(7)] + 10 = \mathbb{E}[N(2)] + 10 = (2.2)(2) + 10 = 14.4$.

(c) Let S_{30} denote the time of occurrence of the 30th event. Independence and stationarity of increments helps us again to compute, for instance, the conditional expectation of S_{30} given that 8 events occurred in the first 12 time periods. We write $\mathbb{E}[S_{30} | N(12) = 8] = 12 + \mathbb{E}[\text{time until 22 more events}] = 12 + \mathbb{E}[S_{22}] = 12 + 22/2.2 = 12 + 10 = 22$. \square

PROPOSITION 3.3. Suppose that in a Poisson process $\{N(t), t \geq 0\}$ with rate λ , an event can be either of category 1 with probability p , or of category 2 with probability $1 - p$. Denote by $N_1(t)$ and $N_2(t)$ the number of events of category 1 and 2 that occur by time t , respectively. Note that $N(t) = N_1(t) + N_2(t)$. Then, $\{N_1(t), t \geq 0\}$ and $\{N_2(t), t \geq 0\}$ are independent Poisson processes with respective rates λp and $\lambda(1 - p)$.

PROOF: Note that by definition, for an observed value of $N(t)$, $N_1(t)$ has a binomial distribution with parameters $N(t)$ and p . Respectively, $N_2(t)$ has a binomial distribution with parameters $N(t)$ and $1 - p$. Therefore, the joint probability distribution of $N_1(t)$ and $N_2(t)$ can be derived as

$$\begin{aligned} & \mathbb{P}(N_1(t) = n_1, N_2(t) = n_2) \\ &= \mathbb{P}(N_1(t) = n_1, N_2(t) = n_2 | N(t) = n_1 + n_2) \mathbb{P}(N(t) = n_1 + n_2) \\ &= \binom{n_1 + n_2}{n_1} p^{n_1} (1 - p)^{n_2} \frac{(\lambda t)^{n_1 + n_2}}{(n_1 + n_2)!} e^{-\lambda t} \\ &= \frac{(\lambda p t)^{n_1}}{n_1!} e^{-\lambda p t} \frac{(\lambda(1 - p)t)^{n_2}}{n_2!} e^{-\lambda(1 - p)t}. \end{aligned}$$

Hence, $N_1(t)$ and $N_2(t)$ are independent Poisson random variables with rates λp and $\lambda(1 - p)$, respectively. Independence and stationarity of increments are inherited from those of the process $\{N(t), t \geq 0\}$. \square

REMARK. In the above proposition, the Poisson process $\{N(t), t \geq 0\}$ is called the *superposition* of Poisson processes $\{N_1(t), t \geq 0\}$ and $\{N_2(t), t \geq 0\}$. In turn, $\{N_1(t), t \geq 0\}$ and $\{N_2(t), t \geq 0\}$ are called *thinned* (or *splitted*) Poisson processes.

EXAMPLE 3.4. Suppose phone calls to a customer service department in a credit card company arrive as a Poisson process with rate 3 per minute. Thirty percent of the calling customers experience technical difficulties when using their credit cards.

(a) The probability that during the next 15 minutes there will be 12 phone calls from customers who experience technical difficulties is computed as follows. We focus only on the customers who experience technical difficulties. Call this process $\{N_1(t), t \geq 0\}$. We know that it is a Poisson process with the rate $\lambda p = (3)(0.30) = 0.9$ per minute. So, we can write $\mathbb{P}(N_1(15) = 12) = \frac{((0.9)(15))^{12}}{12!} e^{-(0.9)(15)} = 0.10488$.

(b) Suppose now we want to calculate the probability that during the next 15 minutes there will be 40 phone calls from customers, half of whom experience technical difficulties. We denote by $\{N_2(t), t \geq 0\}$ the Poisson process that counts only the callers who don't experience technical difficulties. Its rate is $\lambda(1 - p) = (3)(0.7) = 2.1$. We know that the processes $\{N_1(t), t \geq 0\}$ and $\{N_2(t), t \geq 0\}$ behave independently. Hence, we write $\mathbb{P}(N_1(15) = 20, N_2(15) = 20) = \mathbb{P}(N_1(15) = 20) \mathbb{P}(N_2(15) = 20) = \frac{((0.9)(15))^{20}}{20!} e^{-(0.9)(15)} \cdot \frac{((2.1)(15))^{20}}{20!} e^{-(2.1)(15)} = (0.0228)(0.00794) = 0.000181$. \square

3.2 Simulations in R

We will present two simulation methods of a trajectory of a Poisson process.

SIMULATION 3.1. (EXPONENTIAL INTERARRIVALS). When simulating a trajectory of a Poisson process, we need first to simulate exponentially distributed interarrival times. We base our simulations on standard uniform random variables and use the inversion of the cumulative distribution function method to obtain exponentially distributed random variables: if $u \sim \text{Unif}(0, 1)$, then $-\frac{1}{\lambda} \ln(1 - u)$ is exponential with mean $1/\lambda$. Note that since $1 - u$ is also $\text{Unif}(0, 1)$, we can simplify the expression for the exponential random variables to $-\frac{1}{\lambda} \ln u$.

To plot the simulated trajectory, we use the `segment()` function which takes as arguments vectors of left and right endpoints. Below we present the code and graph of a trajectory of a Poisson process with rate 2 that stops when it makes the 20th jump.

```
#specifying parameters
lambda<- 2
njumps<- 20

#defining states
N<- 0:njumps
```

```

#setting time as vector
time<- c()

#setting initial value for time
time[1]<- 0

#specifying seed
set.seed(333422)

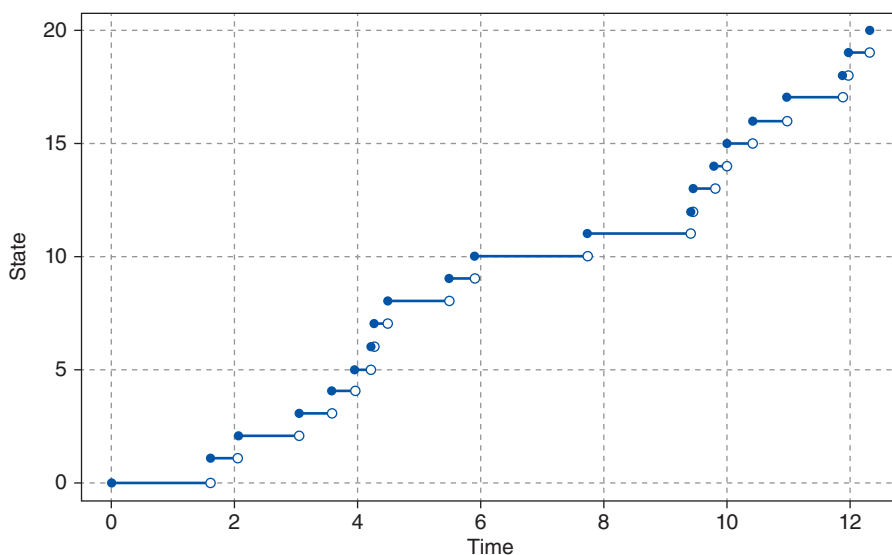
#simulating trajectory
for (i in 2:(njumps+1))
time[i]<- time[i-1]+round((-1/lambda)*log(runif(1)),2)

#plotting trajectory
# type="n" draws empty frame with no graph
plot(time, N, type="n", xlab="Time", ylab="State",
panel.first = grid())

segments(time[-length(time)], N[-length(time)],
time[-1]-0.07, N[-length(time)], lwd=2, col="blue")

points(time, N, pch=20, col="blue")
points(time[-1], N[-length(time)], pch=1, col="blue")

```



□

SIMULATION 3.2. (UNIFORM ORDER STATISTICS). First we need to prove the theoretical result. Consider a Poisson process $\{N(t), t \geq 0\}$. Given that n events occurred by time t , the times at which the events occurred are distributed as the order statistics of a uniform distribution on $(0, t)$.

To show this result, we derive the conditional density of n waiting times S_1, S_2, \dots, S_n , using independence and exponential distribution of the inter-arrival times T_1, T_2, \dots, T_n . We write

$$\begin{aligned} & f_{S_1, S_2, \dots, S_n}(s_1, s_2, \dots, s_n \mid N(t) = n) \\ &= \frac{f_{T_1}(s_1) f_{T_2}(s_2 - s_1) \cdots f_{T_n}(s_n - s_{n-1}) \mathbb{P}(T_{n+1} > t - s_n)}{\mathbb{P}(N(t) = n)} \\ &= \frac{\lambda e^{-\lambda s_1} \lambda e^{-\lambda(s_2 - s_1)} \cdots \lambda e^{-\lambda(s_n - s_{n-1})} e^{-\lambda(t - s_n)}}{\frac{(\lambda t)^n}{n!} e^{-\lambda t}} = \frac{n!}{t^n}. \end{aligned}$$

This gives us the following algorithm to generate trajectories:

- Step 1. Fix t and generate $N(t) \sim Poi(\lambda t)$.
- Step 2. Generate $N(t)$ standard uniform random variables $U_1, \dots, U_{N(t)}$.
- Step 3. Order $U_1, \dots, U_{N(t)}$ in increasing order, obtaining the ordered set $U_{(1)}, \dots, U_{(N(t))}$.
- Step 4. Multiply the order statistics by t to obtain the set of event times $S_1 = tU_{(1)}, \dots, S_{N(t)} = tU_{(N(t))}$.
- Step 5. Define the states of the Poisson process as $N(0) = 0, N(S_1) = 1, N(S_2) = 2, \dots, N(S_{N(t)}) = N(t)$.
- Step 6. Plot the states against time.

The sample code and plot follow.

```
#specifying parameters
t<- 10
lambda<- 2

#specifying seed
set.seed(32114)

#generating N(t)
n_jumps<- rpois(1,lambda*t)
```

```

#defining states
N<- 0:njumps

#generating N(t) standard uniforms
u<- c()
u[1]<- 0

for(i in 2:(njumps+1))
u[i]<- runif(1)

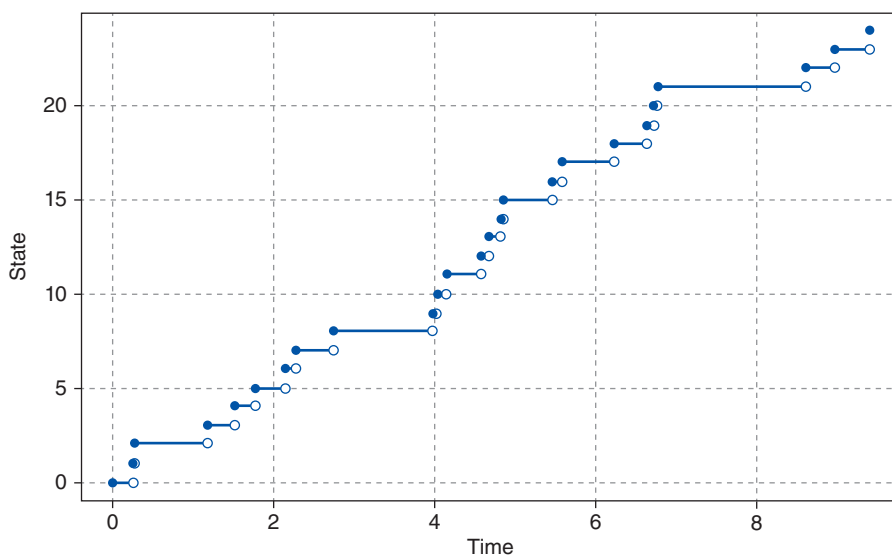
#computing event times
time<- t*sort(u)

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State",
panel.first = grid())

segments(time[-length(time)], N[-length(time)],
time[-1]-0.07, N[-length(time)], lwd=2, col="blue")

points(time, N, pch=20, col="blue")
points(time[-1], N[-length(time)], pch=1, col="blue")

```



□

3.3 Applications of Poisson Process

APPLICATION 3.1. In seismology, occurrence of earthquakes is often modeled according to a Poisson process. We obtain the data from the Southern California Earthquake Data Center's website https://service.scedc.caltech.edu/eq-catalogs/date_mag_loc.php. The data are on earthquakes in Southern California with a minimum magnitude of 3.0 that occurred between 2012 and 2018. We compute the lengths of the interarrival times and remove those earthquakes that were registered within three hours of their predecessors (possibly aftershocks). Finally, we conducted a chi-squared goodness-of-fit test to see if these times follow an exponential distribution. The R code and output follow.

```
eq.data<- read.csv(file="./earthquakedata2012-2018.csv",
header=TRUE, sep=",")

#creating date-time variable
datetime<- as.POSIXct(paste(as.Date(eq.data$DATE),
eq.data$TIME))

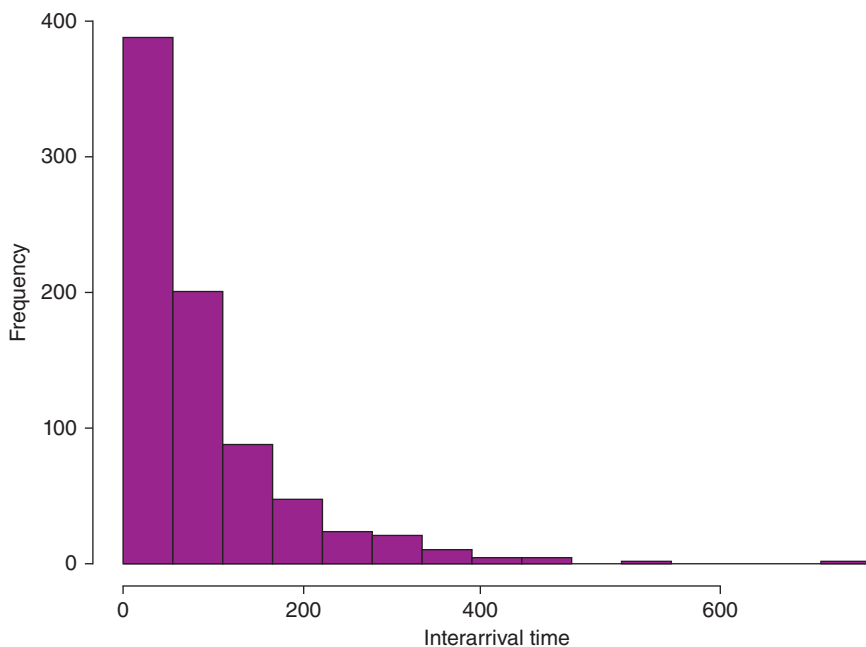
#computing lag
datetime.lag<- c(0,head(datetime, -1))

#computing interarrival times (in hours)
int.time<- (as.numeric(datetime)-as.numeric(datetime.lag))/
3600

#removing first value
int.time<- int.time[-1]

#removing immediate aftershocks (within 3 hours)
int<- int.time[int.time>3]

#plotting histogram
hist(int, main="", col="dark magenta", xlab="Interarrival
Time")
```



```
#binning interarrival times
binned.int<- as.factor(ifelse(int<40,"1",
ifelse(int>=40 & int<80,"2",ifelse(int>=80 & int<120,"3",
ifelse(int>=120 & int<160,"4",ifelse(int>=160 &
int<200,"5",
ifelse(int>=200 & int<240,"6","7"))))))))

#computing observed frequencies
obs<- table(binned.int)

#estimating mean for exponential distribution
mean.est<- mean(int)

#computing expected frequencies
exp<- c(1:7)
exp[1]<- length(int)*(1-exp(-40/mean.est))
exp[2]<- length(int)*(exp(-40/mean.est)-exp(-80/mean.est))
exp[3]<- length(int)*(exp(-80/mean.est)-exp(-120/mean.est))
exp[4]<- length(int)*(exp(-120/mean.est)-exp(-160/mean.est))
exp[5]<- length(int)*(exp(-160/mean.est)-exp(-200/mean.est))
exp[6]<- length(int)*(exp(-200/mean.est)-exp(-240/mean.est))
exp[7]<- length(int)*exp(-240/mean.est)

obs
```

1	2	3	4	5	6	7
342	178	117	49	39	24	42

```
round(exp,1)
```

```
319.8 190.5 113.5 67.6 40.3 24.0 35.4
```

```
#computing chi-squared statistic
print(chi.sq<- sum((obs-exp)^2/exp))
```

```
8.883823
```

```
#computing p-value
print(p.value<- 1-pchisq(chi.sq, df=5))
```

```
0.1137888
```

The number of degrees of freedom in this test is calculated as the number of bins minus 1 and minus the number of parameters that have to be estimated (in this case one mean), so we get that $df = 7 - 1 - 1 = 5$. The p -value is larger than 0.05, indicating that the earthquakes in the given time frame occurred according to a Poisson process. \square

APPLICATION 3.2. In sports analytics, a Poisson process is used to model the process of goal scoring in a game. Consider a team game where players score only one point at a time, for instance, ice hockey. Suppose the points scored by team A follow a Poisson process $\{N_A(t), t \geq 0\}$ with rate λ_A , and points scored by team B are governed by a Poisson process $\{N_B(t), t \geq 0\}$ with parameter λ_B . Assuming the two processes are independent, we can derive some interesting results.

(a) The sum of the two independent Poisson processes $N(t) = N_A(t) + N_B(t)$ is the superposition Poisson process with rate $\lambda_A + \lambda_B$. It represents the process of scoring by either team A or B. If on average, fans wait for time $1/\lambda_A$ for team A to score, respectively, $1/\lambda_B$ for team B to score, then the officials wait, on average, for a shorter period of time $1/(\lambda_A + \lambda_B)$ for either team to score.

To see how it works with numbers, suppose team A scores, on average, every 10 minutes, and team B scores every 12 minutes, on average. Then the expected waiting time until any team scores is $1/(1/10 + 1/12) = 120/22 = 5.45$ seconds.

(b) We can find the probability that one team scores ahead of the other team.

Denote by T_A and T_B the respective interarrival times. We know that T_A and T_B are independent and exponentially distributed with means $\mathbb{E}(T_B) = 1/\lambda_A$ and $\mathbb{E}(T_B) = 1/\lambda_B$. We write

$$\begin{aligned} \mathbb{P}(\text{team } A \text{ scores before team } B) &= \mathbb{P}(T_A < T_B) \\ &= \int_0^\infty \mathbb{P}(T_B > t) f_A(t) dt = \int_0^\infty e^{-\lambda_B t} \lambda_A e^{-\lambda_A t} dt = \frac{\lambda_A}{\lambda_A + \lambda_B}. \end{aligned}$$

Now switching λ_A and λ_B , we get

$$\mathbb{P}(\text{team } B \text{ scores before team } A) = \mathbb{P}(T_B < T_A) = \frac{\lambda_B}{\lambda_A + \lambda_B}.$$

With our numbers, $\mathbb{P}(\text{team } A \text{ scores before } B) = \frac{1/10}{1/10+1/12} = 12/22 = 0.545$, and $\mathbb{P}(\text{team } B \text{ scores before } A) = 1 - 0.545 = 0.455$.

(c) We can find the probability of a tie at the end of the game, and also the probability that team A (team B) wins.

Let T denote the length of the game. Using independence of the two processes, $\{N_A(t), t \geq 0\}$ and $\{N_B(t), t \geq 0\}$, and expressions for the probability mass functions, we write

$$\begin{aligned} \mathbb{P}(\text{game ties}) &= \sum_{n=0}^{\infty} \mathbb{P}(N_A(T) = n, N_B(T) = n) \\ &= \sum_{n=0}^{\infty} \mathbb{P}(N_A(T) = n) \mathbb{P}(N_B(T) = n) \\ &= \sum_{n=0}^{\infty} \frac{(\lambda_A T)^n}{n!} e^{-\lambda_A T} \cdot \frac{(\lambda_B T)^n}{n!} e^{-\lambda_B T} = e^{-(\lambda_A + \lambda_B)T} \sum_{n=0}^{\infty} \frac{(\lambda_A \lambda_B T^2)^n}{(n!)^2}, \\ \mathbb{P}(\text{team } A \text{ wins}) &= \mathbb{P}(N_A(T) > N_B(T)) \\ &= \sum_{n=0}^{\infty} \sum_{k=1}^{\infty} \mathbb{P}(N_A(T) = n+k, N_B(T) = n) \\ &= \sum_{n=0}^{\infty} \sum_{k=1}^{\infty} \mathbb{P}(N_A(T) = n+k) \mathbb{P}(N_B(T) = n) \\ &= \sum_{n=0}^{\infty} \sum_{k=1}^{\infty} \frac{(\lambda_A T)^{n+k}}{(n+k)!} e^{-\lambda_A T} \cdot \frac{(\lambda_B T)^n}{n!} e^{-\lambda_B T} \\ &= e^{-(\lambda_A + \lambda_B)T} \sum_{n=0}^{\infty} \left[\frac{(\lambda_A \lambda_B T^2)^n}{n!} \cdot \sum_{k=1}^{\infty} \frac{(\lambda_A T)^k}{(n+k)!} \right], \end{aligned}$$

and, switching λ_A and λ_B , we get

$$\mathbb{P}(\text{team } B \text{ wins}) = e^{-(\lambda_A + \lambda_B)T} \sum_{n=0}^{\infty} \left[\frac{(\lambda_A \lambda_B T^2)^n}{n!} \cdot \sum_{k=1}^{\infty} \frac{(\lambda_B T)^k}{(n+k)!} \right].$$

The duration of the playing time in an ice hockey game is $T = 60$ minutes. Therefore, we obtain

$$\begin{aligned} \mathbb{P}(\text{game ties}) &= e^{-(1/10+1/12)(60)} \sum_{n=0}^{\infty} \frac{((1/10)(1/12)(60)^2)^n}{(n!)^2} \\ &= e^{-11} \sum_{n=0}^{\infty} \frac{30^n}{(n!)^2} = 0.1166. \end{aligned}$$

We calculated the sum numerically in R. The sum converges after 15 terms.

```
sum<- 0
for(n in 0:15)
sum<- sum+30^n/(factorial(n))^2

sum*exp(-11)
```

0.1165575

Further,

$$\mathbb{P}(\text{team } A \text{ wins}) = e^{-11} \sum_{n=0}^{\infty} \left[\frac{30^n}{n!} \cdot \sum_{k=1}^{\infty} \frac{6^k}{(n+k)!} \right] = 0.5590.$$

R code given below computes this double sum numerically.

```
sum.n<- 0
for (n in 0:15) {
sum.k<-0
  for (k in 1:15)
    sum.k<- sum.k+6^k/factorial(n+k)
sum.n<- sum.n + 30^n/factorial(n)*sum.k
}

sum.n*exp(-11)
```

0.5589743

Finally,

$$\mathbb{P}(\text{team } B \text{ wins}) = e^{-11} \sum_{n=0}^{\infty} \left[\frac{30^n}{n!} \cdot \sum_{k=1}^{\infty} \frac{5^k}{(n+k)!} \right] = 0.3244,$$

as the R code below computes

```
sum.n<- 0
for (n in 0:15) {
  sum.k<-0
  for (k in 1:15)
    sum.k<- sum.k+5^k/factorial(n+k)
  sum.n<- sum.n + 30^n/factorial(n)*sum.k
}

sum.n*exp(-11)
```

0.3244495

Note that the three probabilities add up to 1, as they should be. \square

APPLICATION 3.3. One famous application of a Poisson process is that of a pedestrian versus traffic flow. A pedestrian needs to get to the other side of a road. Assume that cars pass according to a Poisson process with rate λ .

(a) If it takes the pedestrian time τ to cross the road, how long, on average, will it take the person to get to the other side?

Note that the person has to wait for a gap in traffic of length at least τ before he/she can cross. Denote by T the total time (waiting plus crossing). Suppose the person just approached the road. The Poisson process renews itself at this moment, and thus, the person has to wait an exponential time with a mean $1/\lambda$ for the next car. Call this time T_1 . If $T_1 \geq \tau$, then the person can safely cross the road and $T = \tau$. If, however, $T_1 < \tau$, the person has to wait for T_1 for the first car to pass, and then the process renews itself and the pedestrian would have to wait for an additional time \tilde{T} that has the same distribution at T . Thus, T can be written as

$$T = \begin{cases} \tau, & \text{if } T_1 \geq \tau, \\ T_1 + \tilde{T}, & \text{if } T_1 < \tau, \end{cases}$$

where T_1 is an exponentially distributed random variable with mean $1/\lambda$. Consequently,

$$\mathbb{E}(T) = \tau \mathbb{P}(T_1 \geq \tau) + \int_0^\tau t \lambda e^{-\lambda t} dt + \mathbb{E}(T) \mathbb{P}(T_1 < \tau).$$

This can be rewritten as

$$\begin{aligned}\mathbb{E}(T) \mathbb{P}(T_1 \geq \tau) &= \tau \mathbb{P}(T_1 \geq \tau) + \int_0^\tau t \lambda e^{-\lambda t} dt \\ &= \tau e^{-\lambda \tau} - \tau e^{-\lambda \tau} + \int_0^\tau e^{-\lambda t} dt = \frac{1}{\lambda} (1 - e^{-\lambda \tau}).\end{aligned}$$

From here,

$$\mathbb{E}(T) = \frac{1 - e^{-\lambda \tau}}{\lambda e^{-\lambda \tau}} = \frac{1}{\lambda} (e^{\lambda \tau} - 1).$$

Suppose, on average, a car passes every 20 seconds (that is, $\lambda = 1/20 = 0.05$ cars per second), and the pedestrian needs $\tau = 30$ seconds to cross the road. Thus, on average, it takes the person $\mathbb{E}(T) = \frac{1}{0.05} (e^{(0.05)(30)} - 1) = 69.63$ seconds to cross the road.

(b) How many cars, on average, will pass by before the pedestrian can cross?

Let N be the number of cars that pass by before the person can cross. Then $N \geq n$, if and only if the first n interarrival times are all less than τ . Hence, $\mathbb{P}(N \geq n) = (1 - e^{-\lambda \tau})^n$. We can compute the probability of $N = n$ as

$$\mathbb{P}(N = n) = \mathbb{P}(N \geq n) - \mathbb{P}(N \geq n + 1) = (1 - e^{-\lambda \tau})^n - (1 - e^{-\lambda \tau})^{n+1}.$$

Let $a = 1 - e^{-\lambda \tau}$. The expected value of N can then be computed as

$$\begin{aligned}\mathbb{E}(N) &= \sum_{n=0}^{\infty} n \mathbb{P}(N = n) = \sum_{n=0}^{\infty} n (a^n - a^{n+1}) = a - a^2 + (2)(a^2 - a^3) \\ &\quad + (3)(a^3 - a^4) + \cdots = a + a^2 + a^3 + \cdots = \frac{1}{1-a} - 1 = e^{\lambda \tau} - 1.\end{aligned}$$

Note that $\mathbb{E}(T) = (1/\lambda)\mathbb{E}(N)$. It is intuitively so, because the person has to wait until an average of $\mathbb{E}(N)$ cars pass, and the average waiting time between these cars is $1/\lambda$ seconds. In our numeric example, $\lambda = 0.05$ and $\tau = 30$. So, $\mathbb{E}(N) = e^{(0.05)(30)} - 1 = 3.48$ cars. \square

Exercises

EXERCISE 3.1. Let $\{N(t), t \geq 0\}$ be a Poisson process with rate λ . Find the joint probability distribution $\mathbb{P}(N(s) = m, N(t) = n)$, for any $t \geq s \geq 0$, and $n \geq m \geq 0$.

EXERCISE 3.2. Show that for a Poisson process $\{N(t), t \geq 0\}$ with rate λ , the covariance between $N(s)$ and $N(t)$ is equal to $\lambda \min(s, t)$, for any $s, t \geq 0$.

EXERCISE 3.3. An insurance agent handles policyholders' claims. Claims are submitted on weekdays according to a Poisson process with a rate $\lambda = 5$ per day.

(a) If there were two claims submitted on Monday and three on Tuesday, what is the probability that by the end of the day on Friday there will be a total of 16 claims submitted that week?

(b) In the new calendar year, the agent opens for business on Monday, January 2. On what day does he expect to see the 100th claim?

EXERCISE 3.4. A salesperson contacts customers over the phone and offers his product. Assume that the times that pass between consecutive phone calls (that includes the call and the break in-between) are independent and exponentially distributed with mean of 5 minutes. He estimates that 15% of all the customers he calls actually buy his product.

(a) Calculate the expected number of successful sales in the next 2 hours.

(b) Compute the probability that within 1 hour he places 15 calls, 5 of which result in a sale.

(c) Find the conditional probability that he makes 10 sales in 4 hours, given that he has made 3 sales the first hour.

EXERCISE 3.5. People contract a disease according to a Poisson process with an unknown rate λ . Suppose the incubation period until symptoms of the disease show is a random variable with a known cumulative distribution function F . Let $N_1(t)$ denote the number of individuals who have shown symptoms by time t , and let $N_2(t)$ be the number of individuals who have not yet shown any symptoms by time t .

(a) Argue that $\{N_1(t), t \geq 0\}$ and $\{N_2(t), t \geq 0\}$ are independent Poisson processes with means

$$\mathbb{E}(N_1(t)) = \lambda \int_0^t F(u) du \quad \text{and} \quad \mathbb{E}(N_2(t)) = \lambda \int_0^t (1 - F(u)) du.$$

(b) For a known time t and observed number of individuals showing symptoms $\hat{\mathbb{E}}(N_1(t))$, prove that the estimated number of individuals infected but not yet showing symptoms is

$$\hat{\mathbb{E}}(N_2(t)) = \frac{\hat{\mathbb{E}}(N_1(t)) \int_0^t (1 - F(u)) du}{\int_0^t F(u) du}.$$

(c) Suppose the incubation period until symptoms show is an exponentially distributed random variable with a mean of 2 days. If 1,000 individuals show

symptoms of a disease by day 10, estimate the number of individuals who are also infected but haven't shown the symptoms yet.

EXERCISE 3.6. Areas of high road surface distress (potholes or cracks) that need immediate attention of road maintenance operators are distributed according to a Poisson law with a rate of 2.8 per mile.

- (a) What is the average number of distressed road surface areas on a 10-mile stretch of a freeway?
- (b) Simulate locations of 30 distressed surface areas. What is the total length of the road in your simulation?
- (c) Suppose there are 30 distressed surface areas on a 10-mile stretch of a freeway. Simulate locations of those areas.

EXERCISE 3.7. The National Geophysical Data Center's website

<https://www.ngdc.noaa.gov/hazel/view/hazards/volcano/event-search/>

provides access to the Global Significant Volcanic Eruptions Database. Verify that those volcanic eruptions in the past 100 years can be modeled as a Poisson process.

EXERCISE 3.8. Two teams are playing basketball. Team A opportunities to score appear as a Poisson process with a rate of 0.5 per minute. Suppose that 25% bring one point, 40% bring two points, 20% bring the team three points, and the others result in missed shots. For team B, the opportunities come as a Poisson process with a rate of 0.4 per minute, of which 25% result in a 1-pointer, 50% result in a 2-pointer, 15% result in a 3-pointer, and the rest are missed.

- (a) How long, on average, does the stadium have to wait until a team scores?
- (b) How long, on average, will the fans wait until team A scores? Team B scores?
- (c) What is the probability that team A scores before team B? Team B scores before team A?
- (d) What is the probability that at the end of the 48-minute game, teams A and B will score the same number of 1-pointers, the same number of 2-pointers, and the same number of 3-pointers?

EXERCISE 3.9. In a popular nursery rhyme,

*Itsy bitsy spider went up the water spout.
Down came the rain and washed the spider out.
Out came the sun and dried up all the rain,
And the itsy bitsy spider went up the spout again.*

Assume that the length of the downspout is 30 feet, and the spider climbs with a constant speed of 1 foot per minute. The rain comes down as a Poisson process with a rate of 2 per hour.

- (a) Find the expected time it takes the spider to reach the top.
- (b) Find the expected number of times the spider will be washed down before it reaches the top.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Nonhomogeneous Poisson Process

4.1 Definition of Nonhomogeneous Poisson Process

The Poisson process considered in the previous chapter has a constant rate λ . In some situations, it is difficult to assume that the rate doesn't change over a large period of time. In this case, we can make λ depend on time t and define a Poisson process with rate $\lambda(t)$. The rate is now called the *intensity rate* or *intensity function*. The process still starts at zero at time zero, and its increments are still independent, but now the increments are non-stationary.

A counting process $\{N(t), t \geq 0\}$ is called a *nonhomogeneous* (or *non-stationary*, or *time-dependent*) Poisson process¹ if: (i) $N(0) = 0$, (ii) increments are independent, and (iii) for all $s, t \geq 0$,

$$\mathbb{P}(N(t+s) - N(s) = n) = \frac{\left(\int_s^{t+s} \lambda(u) du\right)^n}{n!} e^{-\left(\int_s^{t+s} \lambda(u) du\right)}, \quad n \geq 0.$$

Define the function $\Lambda(t) = \int_0^t \lambda(u) du$. It is called the *integrated intensity rate function* or the *mean value function*. The probability mass function of a nonhomogeneous Poisson process can be written in terms of $\Lambda(t)$ as

$$\mathbb{P}(N(t+s) - N(s) = n) = \frac{[\Lambda(t+s) - \Lambda(s)]^n}{n!} e^{-[\Lambda(t+s) - \Lambda(s)]}, \quad n \geq 0.$$

Note that for $s, t \geq 0$, $\mathbb{E}(N(t+s) - N(s)) = \Lambda(t+s) - \Lambda(s)$.

REMARK 4.1. Here we formulate an alternative definition of a nonhomogeneous Poisson process. It can be shown that the two definitions are equivalent. We will need this definition to justify the method we use in Simulation 4.3. A counting process $\{N(t), t \geq 0\}$ is termed a *nonhomogeneous* Poisson process with the intensity rate $\lambda(t)$, $t \geq 0$, if: (i) $N(0) = 0$, (ii) increments are

¹Introduced by a prominent statistician Sir David Roxbee Cox in his 1955 paper "Some Statistical Methods Connected with Series of Events." *Journal of the Royal Statistical Society. Series B (Methodological)*, 17(2): 129 – 164.

independent and stationary, (iii) for any fixed $t \geq 0$ and any small positive increment Δt , $\mathbb{P}(\text{no events happen in } [t, t + \Delta t]) = 1 - \lambda(\Delta t) + o(\Delta t)$, and (iv) $\mathbb{P}(\text{one event happens in } [t, t + \Delta t]) = \lambda(\Delta t) + o(\Delta t)$, where $o(\Delta t)$ (pronounced “little oh of delta t”) denotes any function of Δt that goes to zero faster than Δt . That is, $o(\Delta t) = \{f(\Delta t) : \lim_{\Delta t \rightarrow 0} \frac{f(\Delta t)}{\Delta t} = 0\}$. \square

EXAMPLE 4.1. Throughout the day, arrivals of phone calls to a doctor’s office can be modeled as a nonhomogeneous Poisson process with the intensity rate

$$\lambda(t) = \begin{cases} 10, & \text{if } 9\text{AM} \leq t \leq 10:30\text{AM}, \\ 5, & \text{if } 10:30\text{AM} < t \leq 12\text{PM}, \\ 8, & \text{if } 12\text{PM} < t \leq 1\text{PM}, \\ 4, & \text{if } 1\text{PM} < t \leq 5\text{PM}. \end{cases}$$

(a) To calculate the integrated rate function $\Lambda(t)$, we need to define the time variable as ranging between 0 and 8 hours of the workday. We can rewrite the intensity rate as

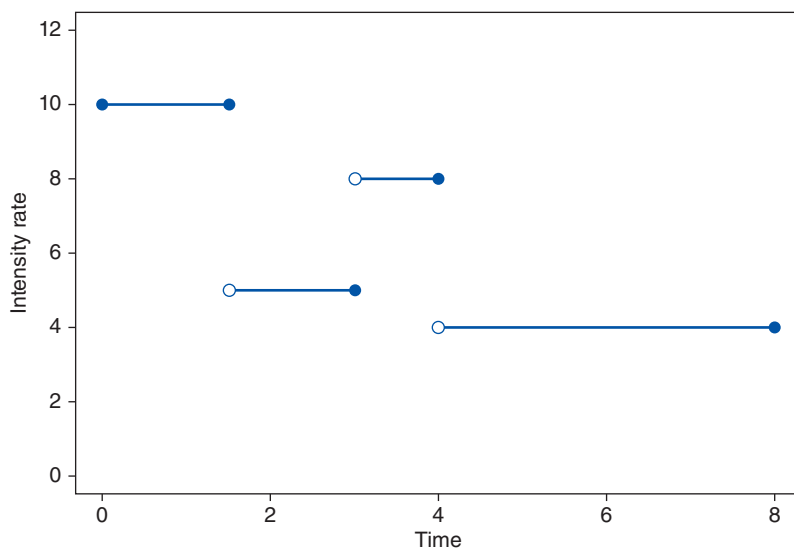
$$\lambda(t) = \begin{cases} 10, & \text{if } 0 \leq t \leq 1.5, \\ 5, & \text{if } 1.5 < t \leq 3, \\ 8, & \text{if } 3 < t \leq 4, \\ 4, & \text{if } 4 < t \leq 8. \end{cases}$$

The integrated rate function is then computed as

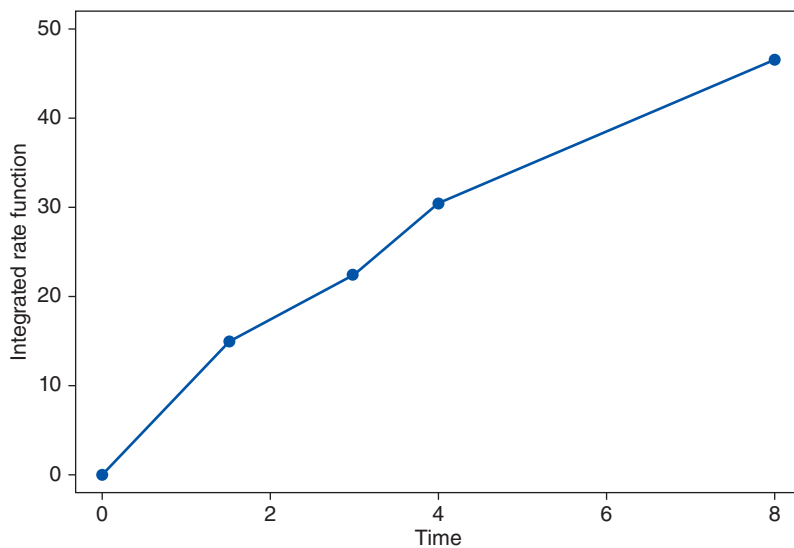
$$\Lambda(t) = \int_0^t \lambda(u) du = \begin{cases} \int_0^t 10 du = 10t, & \text{if } 0 \leq t \leq 1.5, \\ 15 + \int_{1.5}^t 5 du = 15 + 5(t - 1.5), & \text{if } 1.5 < t \leq 3, \\ 22.5 + \int_3^t 8 du = 22.5 + 8(t - 3), & \text{if } 3 < t \leq 4, \\ 30.5 + \int_4^t 4 du = 30.5 + 4(t - 4), & \text{if } 4 < t \leq 8. \end{cases}$$

(b) Below we plot both functions. The R codes are provided.

```
#plotting intensity rate
t=c(0,1.5,3,4,8)
lambda=c(10, 10,5,8,4)
plot(t, lambda, type="n", col="blue ", xlim=c(0,8),
ylim=c(0,12), xlab="Time", ylab="Intensity rate")
segments(t[-5]+0.07, lambda[-1], t[-1], lambda[-1], lwd=2,
col="blue")
points(t, lambda, cex=1.2, pch=19, col="blue")
points(t[-5], lambda[-1], cex=1.2, pch=1, col="blue")
```



```
#plotting integrated rate function
t<- c(0, 1.5, 3, 4, 8)
Lambda<- c(0, 15, 22.5, 30.5, 46.5)
plot(t,Lambda, type="l", lwd=2, col="blue", xlim=c(0,8),
ylim=c(0,50), xlab="time", ylab="integrated rate function")
points(t, Lambda, cex=1.2, pch=16, col="blue")
```



(c) Suppose we want to compute the probability that there will be 15 phone calls between 11AM and 2PM. The time 11AM corresponds to 2 hours and 2PM corresponds to 5 hours after the office opens. We write

$$\begin{aligned}\mathbb{P}(N(5) - N(2) = 15) &= \frac{[\Lambda(5) - \Lambda(2)]^{15}}{15!} e^{-[\Lambda(5) - \Lambda(2)]} \\ &= \frac{(30.5 + 4(5 - 4) - (15 + 5(2 - 1.5)))^{15}}{15!} e^{-(30.5 + 4(5 - 4) - (15 + 5(2 - 1.5)))} \\ &= \frac{(17)^{15}}{15!} e^{-17} = 0.011468.\end{aligned}$$

(d) Finally, we want to compute the average number of phone calls per day. We compute

$$\mathbb{E}(N(8) - N(0)) = \Lambda(8) - \Lambda(0) = 30.5 + 4(8 - 4) - 0 = 46.5 \text{ calls.} \quad \square$$

4.2 Simulations in R

Recall that in [Section 3.2](#) we discussed two simulation methods for trajectories of a homogeneous Poisson process. In the present section, we generalize these methods to the case of a nonhomogeneous Poisson process.

SIMULATION 4.1. (EXPONENTIAL INTERARRIVALS). In this method, we simulate interarrival times. In the nonhomogeneous case, the distributions of interarrival times are not independent. They are obtained as follows.

The first interarrival time has the cumulative distribution function $F_{T_1}(t) = 1 - e^{-\Lambda(t)}$, $t \geq 0$. For a fixed time of the first event occurrence $S_1 = T_1 = s_1$, the cumulative distribution function of T_2 is

$$F_{T_2|S_1}(t|s_1) = 1 - e^{-(\Lambda(t+s_1) - \Lambda(s_1))}, \quad t \geq 0.$$

In general, for a given waiting time for the n th event $S_n = s_n$, the conditional distribution of the interarrival time T_{n+1} is

$$F_{T_{n+1}|S_n}(t|s_n) = 1 - e^{-(\Lambda(t+s_n) - \Lambda(s_n))}, \quad t \geq 0, \quad n \geq 1.$$

As an example, here we give the code that simulates a trajectory of a nonhomogeneous Poisson process with the integrated rate function $\Lambda(t) = t + 0.05t^2$, $t \geq 0$. In the code we first generate standard uniform random variables $U_i, i = 1, \dots, n$, and then compute event times by inverting the cumulative distribution function. We write $1 - e^{-(S_1 + 0.05S_1^2)} = U_1$, which

solution is $S_1 = \sqrt{100 - 20 \ln(1 - U_1)} - 10$. It can be simplified by replacing $1 - U_1$ by U_1 since both are standard uniform random variables. Thus we have $S_1 = \sqrt{100 - 20 \ln(U_1)} - 10$.

The second event time S_2 solves $1 - e^{-[S_2 + 0.05 S_2^2 - (S_1 + 0.05 S_1^2)]} = U_2$, or equivalently, $(S_2 + 10)^2 - (S_1 + 10)^2 = -20 \ln(1 - U_2)$. The solution is (with $1 - U_2$ replaced by U_2) $S_2 = \sqrt{(S_1 + 10)^2 - 20 \ln(U_2)} - 10$. The general recurrence formula is $S_{n+1} = \sqrt{(S_n + 10)^2 - 20 \ln(U_{n+1})} - 10$. We continue generating the event times according to this formula until we reach a pre-specified number of events of the process. The code and graph are given below.

```
#specifying parameters
njumps<- 20

#defining states
N<- 0:njumps

#defining times as vectors
time<- c()

#specifying seed
set.seed(76855)

#generating standard uniforms
u<- c()
for(i in 1:njumps)
u[i]<- runif(1)

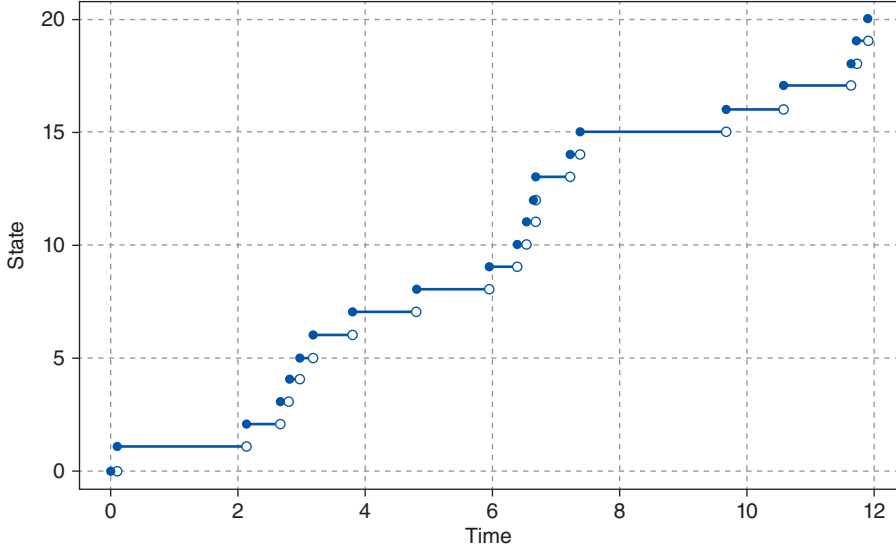
#computing event times
time[1]<- 0
time[2]<- sqrt(100-20*log(u[1]))-10

for(i in 3:(njumps+1)) {
time[i]<- sqrt((time[i-1]+10)^2-20*log(u[i-1]))-10
}

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State",
panel.first=grid())

segments(time[-length(time)], N[-length(time)],
time[-1]-0.07, N[-length(time)], lwd=2, col="blue")

points(time, N, pch=20, col="blue")
points(time[-1], N[-length(time)], pch=1, col="blue")
```

□

SIMULATION 4.2. (UNIFORM ORDER STATISTICS). Let $\{N(t), t \geq 0\}$ denote a nonhomogeneous Poisson process with the intensity rate $\lambda(t)$, $t \geq 0$, and integrated rate function $\Lambda(t)$, $t \geq 0$. Given that n events occurred by time t , the conditional joint density of n waiting times S_1, S_2, \dots, S_n , is derived as

$$\begin{aligned}
 & f_{S_1, S_2, \dots, S_n}(s_1, s_2, \dots, s_n \mid N(t) = n) \\
 &= \frac{f_{T_1}(s_1) f_{T_2}(s_2 - s_1) \cdots f_{T_n}(s_n - s_{n-1}) \mathbb{P}(T_{n+1} > t - s_n)}{\mathbb{P}(N(t) = n)} \\
 &= \frac{\lambda(s_1) e^{-\Lambda(s_1)} \lambda(s_2 - s_1) e^{-(\Lambda(s_2) - \Lambda(s_1))} \cdots \lambda(s_n - s_{n-1}) e^{-(\Lambda(s_n) - \Lambda(s_{n-1}))}}{\frac{(\Lambda(t))^n}{n!} e^{-\Lambda(t)}} \\
 &= n! \frac{\lambda(s_1) \lambda(s_2 - s_1) \cdots \lambda(s_n - s_{n-1})}{\Lambda(t)^n}.
 \end{aligned}$$

This means that the event times are order statistics from the distribution with density $f_S(s) = \frac{\lambda(s)}{\Lambda(t)}$, $0 \leq s \leq t$, and the cumulative distribution function $F_S(s) = \frac{\Lambda(s)}{\Lambda(t)}$, $0 \leq s \leq t$.

In our example with $\Lambda(t) = t + 0.05t^2$, $t \geq 0$, to generate a trajectory, we proceed as follows.

- Step 1. Fix t and generate $N(t) \sim Poi(\Lambda(t))$. For instance, for $t = 10$, $\Lambda(10) = 10 + (0.05)(10^2) = 15$, and so, we would generate $N(t) \sim Poi(15)$.
- Step 2. Generate $N(t)$ standard uniform random variables $U_1, \dots, U_{N(t)}$.
- Step 3. Order $U_1, \dots, U_{N(t)}$ in increasing order, obtaining the ordered set $U_{(1)}, \dots, U_{(N(t))}$.
- Step 4. Compute waiting times $S_1, \dots, S_{N(t)}$ that are positive solutions of the equations $\frac{\Lambda(S_i)}{\Lambda(t)} = U_{(i)}$, or $S_i + 0.05 S_i^2 = 15 U_{(i)}$. That is, $S_i = 10\sqrt{1 + 3U_{(i)}} - 10$.
- Step 5. Define the states of the Poisson process as $N(0) = 0, N(S_1) = 1, N(S_2) = 2, \dots, N(S_{N(t)}) = N(t)$.
- Step 6. Plot the states against time.

The code and plot follow.

```
#specifying parameters
t<- 10
Lambda<- t+0.05*t^2

#specifying seed
set.seed(997755)

#generating N(t)
njumps<- rpois(1,Lambda)

#defining states
N<- 0:njumps

#generating N(t) standard uniforms
u<- c()
u[1]<- 0

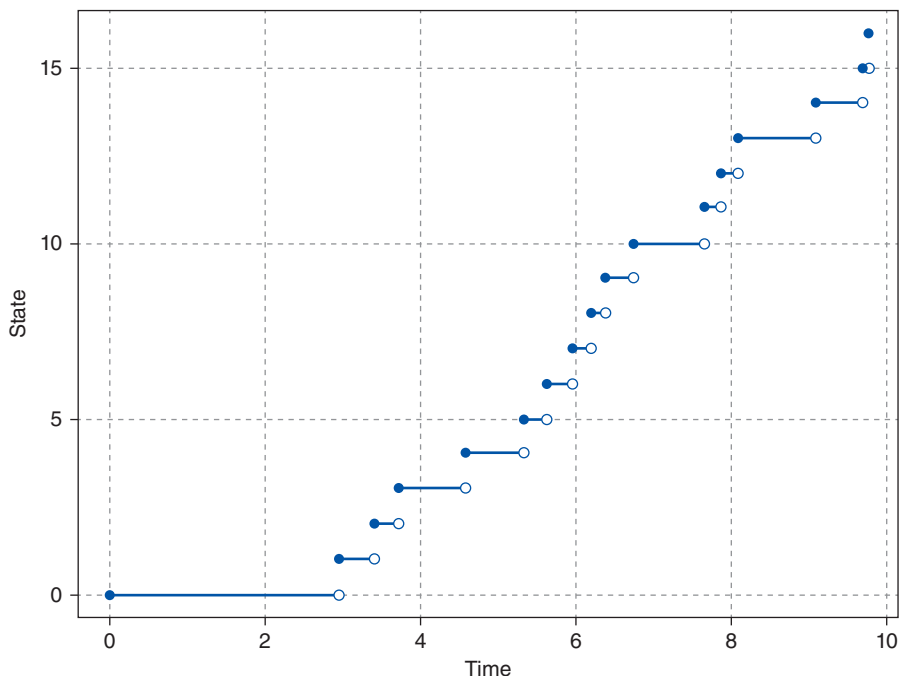
for(i in 2:(njumps+1))
u[i]<- runif(1)

#computing event times
time<- 10*sqrt(1+3*sort(u))-10

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State",
panel.first=grid())
```

```
segments(time[-length(time)], N[-length(time)],
time[-1]-0.07, N[-length(time)], lwd=2, col="blue")

points(time, N, pch=20, col="blue")
points(time[-1], N[-length(time)], pch=1, col="blue")
```



□

Below we introduce a third method of simulating a trajectory of a nonhomogeneous Poisson process. In the case of a homogeneous Poisson process, this method becomes trivial and hence not useful.

SIMULATION 4.3. (THINNING). Let $\{N(t), t \geq 0\}$ denote a nonhomogeneous Poisson process with the intensity rate $\lambda(t)$, $t \geq 0$, and suppose its integrated rate function doesn't have an explicit form or is not easily invertible, so the previous two simulation methods won't work for this process. Here we introduce another method, called the *thinning* method. In this method, first we generate a nonhomogeneous process $\{N^*(t), t \geq 0\}$ with intensity rate $\lambda^*(t)$, $t \geq 0$, that uniformly dominates $\lambda(t)$. That is, $\lambda(t) \leq \lambda^*(t)$ for all $t \geq 0$. The process is selected in such a way that its integrated rate function is invertible, and so $N^*(t)$ can be generated by either of the previous two methods.

Sometimes $\lambda^*(t)$ may be chosen to be a constant, resulting in a homogeneous Poisson process, which is even simpler to generate ([Section 3.2](#)).

Further, once the event times s_1^*, \dots, s_N^* for the process $\{N^*(t), t \geq 0\}$ are generated, they are “thinned” according to the following acceptance-rejection rule: if a standard uniform random variable is less than or equal to the ratio $\lambda(s_i^*)/\lambda^*(s_i^*)$, the time s_i^* is accepted, otherwise, rejected.

The accepted times are the event times for the process $\{N(t), t \geq 0\}$. To show this, we use the alternative definition of a nonhomogeneous Poisson process given in Remark 4.1 and argue as follows. The process $\{N^*(t), t \geq 0\}$ starts at zero and has independent and stationary increments. These properties are inherited by $\{N(t), t \geq 0\}$. Further, in an infinitesimally small interval of length Δt , there are zero occurrences of the process $\{N(t), t \geq 0\}$ if there are no occurrences of the process $\{N^*(t), t \geq 0\}$ or there is one occurrence but it is rejected. There is one occurrence of the process $\{N(t), t \geq 0\}$ if there is one occurrence of the process $\{N^*(t), t \geq 0\}$ and it is accepted. Denoting by T^* an interarrival time, and by U a standard uniform random variable, we write

$$\begin{aligned} \mathbb{P}(N(\Delta t) = 0) &= \mathbb{P}(N^*(\Delta t) = 0) + \mathbb{P}(N^*(\Delta t) = 1) \mathbb{P}\left(U > \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)}\right) \\ &= \mathbb{P}(T^* > \Delta t) + \mathbb{P}(T^* < \Delta t) \mathbb{P}\left(U > \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)}\right) \\ &= e^{-\lambda^*(\Delta t)} + (1 - e^{-\lambda^*(\Delta t)}) \left(1 - \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)}\right) \\ &= 1 - \lambda^*(\Delta t) + o(\Delta t) + (\lambda^*(\Delta t) + o(\Delta t)) \left(1 - \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)}\right) \\ &= 1 - \lambda(\Delta t) + o(\Delta t), \text{ for small } \Delta t, \end{aligned}$$

and

$$\begin{aligned} \mathbb{P}(N(\Delta t) = 1) &= \mathbb{P}(N^*(\Delta t) = 1) \mathbb{P}\left(U \leq \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)}\right) \\ &= \mathbb{P}(T^* < \Delta t) \mathbb{P}\left(U \leq \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)}\right) = (1 - e^{-\lambda^*(\Delta t)}) \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)} \\ &= (\lambda^*(\Delta t) + o(\Delta t)) \frac{\lambda(\Delta t)}{\lambda^*(\Delta t)} = \lambda(\Delta t) + o(\Delta t), \text{ for small } \Delta t. \end{aligned}$$

The above derivation shows that $N(t)$ is a nonhomogeneous Poisson process with the intensity rate function $\lambda(t)$.

Below we present the code that simulates a trajectory of a nonhomogeneous Poisson process with the intensity rate $\lambda(t) = 20 + 20 \sin(\pi t)$, $t \geq 0$. For this

process, the integrated intensity rate function $\Lambda(t) = 20t - \frac{20}{\pi} \cos(\pi t) + \frac{20}{\pi}$, $t \geq 0$, is not readily invertible. However, $\lambda(t) \leq 40$, and so we simulate event times for a Poisson process with rate $\lambda^*(t) = 40$, and then accept an event time s if $U \leq \lambda(s)/\lambda^*(s) = 20(1 + \sin(\pi s))/40 = 0.5(1 + \sin(\pi s))$, and reject otherwise.

```
#specifying parameters
lambda<- function(t) { 20+20*sin(pi*t) }
lambda.star<- function(t) 40
Lambda.star<- function(t) 40*t

#specifying seed
set.seed(2866514)

#generating N(10)
njumps<- rpois(1, Lambda.star(10))

#generating N(10) standard uniforms
u<- c()
u[1]<- 0

for(i in 2:(njumps+1))
u[i]<- runif(1)

#computing event times
time.star<- 10*sort(u)

#thinning event times
accepted<- c()
time<- c()
accepted[1]<- 1
time[1]<- 0

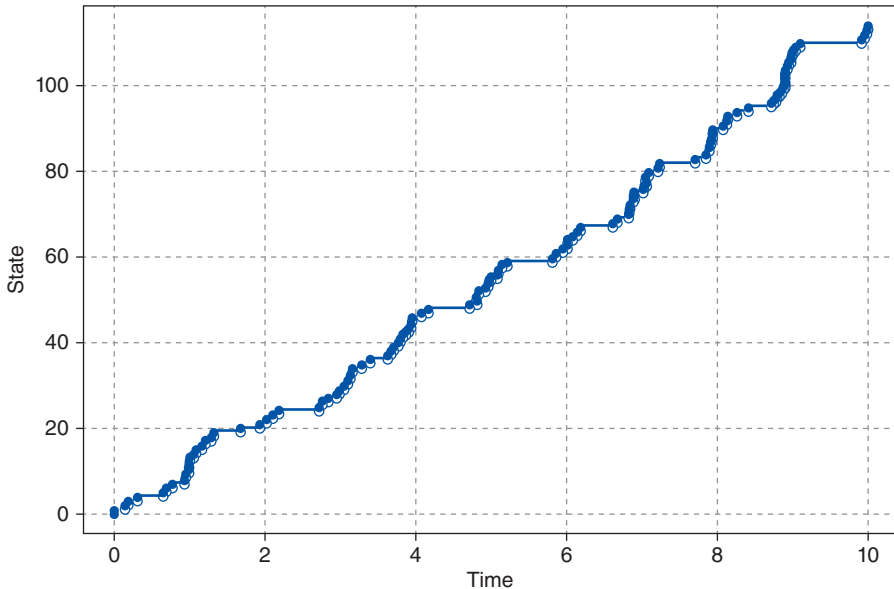
for (i in 2:(njumps+1)) {
if (runif(1)<= lambda(time.star[i])/lambda.star(time.star[i]))
accepted[i]=1 else accepted[i]=0
}

time<- time.star[-which(accepted==0)]
N<- 0:(length(time)-1)

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State",
panel.first = grid())
```

```
segments(time[-length(time)],N[-length(time)], time[-1]-0.07,
N[-length(time)], lwd=2, col="blue")

points(time, N, ylim=c(0,120), pch=20, col="blue")
points(time[-1],N[-length(time)],pch=1, col="blue")
```



□

4.3 Applications of Nonhomogeneous Poisson Process

APPLICATION 4.1. In Application 3.1, we modeled the occurrence of earthquakes in Southern California between 2012 and 2018 via a Poisson model and concluded that it fits the data well. Now we will try to fit a nonhomogeneous Poisson process to a larger data set covering the time span between 2010 and 2020. In this larger time interval, the event rate doesn't stay constant.

First, we estimate the intensity rate function. The code below calculates λ as the ratio of the number of earthquakes per month over the number of days in that month, producing the estimated daily rate for each of the 120 months. One outlying value is removed (for July of 2019), leaving us with 119

values. As the time variable, we compute the accrued number of days from 09/02/2010 to the median day for each of the 119 months. Next, we plot the estimated λ against time and fit a fourth-degree polynomial regression. We then define the intensity rate function $\hat{\lambda}(t)$ as the fourth-degree polynomial with the estimated coefficients. After that, we subdivide the timeline between 09/02/2010 and 08/28/2020 into 9 bins of size 400 days and calculate the observed number of earthquakes in each bin. Then we compute the expected number of occurrences in each bin as the integral of $\hat{\lambda}(t)$ between the lower and upper values of time in this bin. Finally, we compute the chi-squared statistic for the goodness-of-fit test and output the p -value.

The code and all the necessary outputs follow.

```
eq.data<- read.csv(file="./earthquakedata2010-2020.csv",
header=TRUE, sep=",")

#creating date-time variable
eq.data$datetime<- as.POSIXct(paste(as.Date(eq.data$DATE),
eq.data$TIME))

#computing lag
eq.data$datetime.lag<- c(0,head(eq.data$datetime, -1))
#removing first row
eq.data<-eq.data[-1,]

#computing interarrival times (in hours)
eq.data$elapsed.time<- (as.numeric(eq.data$datetime)
-as.numeric(eq.data$datetime.lag))/3600

#removing immediate aftershocks (within 1 hour)
eq.data<- eq.data[eq.data$elapsed.time>1,]

#creating year-month variable
eq.data$year.month<- format(as.Date(eq.data$DATE), "%Y-%m")

#creating unique year-month and number of earthquakes per
month
freq.month<- data.frame(table(eq.data$year.month))
year.month.unique<-freq.month[,1]
neq.month<- freq.month[,2]
neq.month
```

[1]	37	43	29	35	29	19	19	26	18	23	24	11	12	16	19	6
[17]	10	15	10	16	19	12	20	27	14	12	8	10	9	8	8	10
[33]	20	14	7	14	14	12	7	12	8	8	17	9	7	11	21	9
[49]	10	14	5	9	5	10	7	7	14	9	4	2	2	9	5	16
[65]	10	13	7	8	4	13	6	10	17	11	3	10	6	5	10	11
[81]	9	10	10	8	8	8	6	9	9	5	6	11	9	7	7	12
[97]	11	10	9	7	14	11	3	9	5	18	126	35	20	11	25	21
[113]	15	12	12	16	19	30	10	15								

```
#removing outlier (July, 2019, 107th entry)
year.month.unique<- year.month.unique[-107]
neq.month<- neq.month[-107]

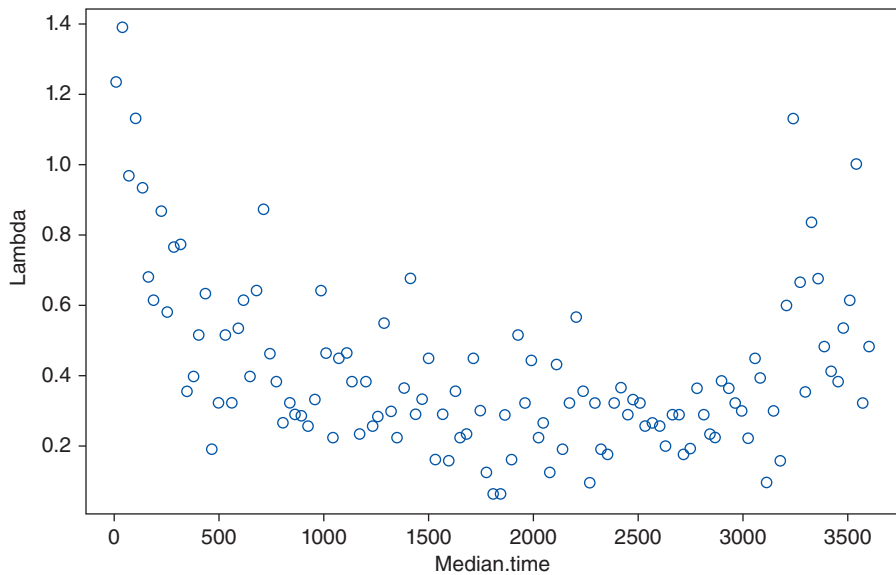
#computing number of days per month
library(lubridate)
day1.month <- ymd(paste(year.month.unique,"01", sep="-") )
library(Hmisc)
ndays.month<- monthDays(as.Date(day1.month, "%Y-%m-%d"))

#estimating intensity rate of earthquakes per day
lambda<- neq.month/ndays.month

#computing cumulative number of days until median day of each
month
median.time<- c()
ndays.total<- c()
median.time[1]<- ndays.month[1]/2
ndays.total[1]<- ndays.month[1]

for (i in 2:length(ndays.month)) {
  median.time[i]<- ndays.total[i-1] + ndays.month[i]/2
  ndays.total[i]<- ndays.total[i-1] + ndays.month[i]
}

#plotting lambda against median time
plot(median.time, lambda)
```

```
#regressing lambda on time
median.time.re<- median.time/1000
median.time.sq<- median.time.re^2
median.time.cu<- median.time.re^3
median.time.qd<- median.time.re^4
glm(lambda~ median.time.re + median.time.sq + median.time.cu
+ median.time.qd)
```

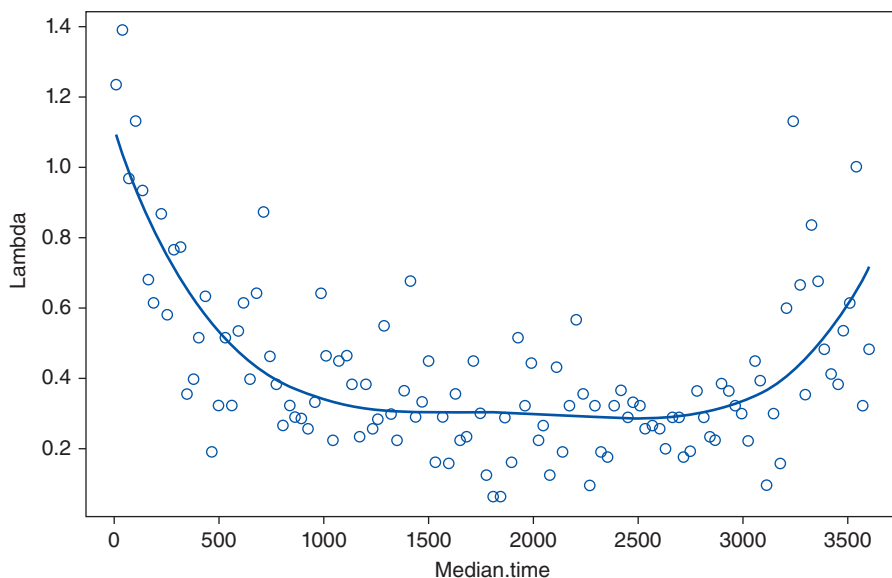
Coefficients:

```
(Intercept)  median.time.re  median.time.sq  median.time.cu
      1.11674         -1.75263         1.41060         -0.49950

median.time.qd
      0.06504
```

```
#adding fitted line
lambda.fn<- function(t) { 1.11674-1.75263*(t/1000)+1.41060*
(t/1000)^2 -0.49950*(t/1000)^3+0.06504*(t/1000)^4 }

lines(median.time, lambda.fn(median.time), lwd=2,
col="blue")
```



```
#conducting goodness-of-fit test
```

```
#binning times
```

```
time.binned<- as.factor(ifelse(as.Date(eq.data$DATE)
<"2011/10/07", "1", ifelse(as.Date(eq.data$DATE)>="2011/10/07"&
as.Date(eq.data$DATE)
<"2012/11/10", "2", ifelse(as.Date(eq.data$DATE)>="2012/11/10"
& as.Date(eq.data$DATE)<"2013/12/15", "3",
ifelse(as.Date(eq.data$DATE)
>="2013/12/15"& as.Date(eq.data$DATE)<"2015/01/19", "4",
ifelse(as.Date(eq.data$DATE)>="2015/01/19"&
as.Date(eq.data$DATE)
<"2016/02/23", "5", ifelse(as.Date(eq.data$DATE)>="2016/02/23"&
as.Date(eq.data$DATE)<"2017/03/29", "6",
ifelse(as.Date(eq.data$DATE)
>="2017/03/29"& as.Date(eq.data$DATE)<"2018/05/03", "7",
ifelse(as.Date(eq.data$DATE)>="2018/05/03"&
as.Date(eq.data$DATE)
<"2019/06/07", "8", "9"))))))))
```

```
#computing observed frequencies
```

```
obs<- table((time.binned))
```

```
#computing expected frequencies
exp<- c()
exp[1]<- integrate(lambda.fn, 0, 400)$value
exp[2]<- integrate(lambda.fn, 400, 800)$value
exp[3]<- integrate(lambda.fn, 800, 1200)$value
exp[4]<- integrate(lambda.fn, 1200, 1600)$value
exp[5]<- integrate(lambda.fn, 1600, 2000)$value
exp[6]<- integrate(lambda.fn, 2000, 2400)$value
exp[7]<- integrate(lambda.fn, 2400, 2800)$value
exp[8]<- integrate(lambda.fn, 2800, 3200)$value
exp[9]<- sum(obs)-sum(exp)
```

```
obs
```

```
  1   2   3   4   5   6   7   8   9
326 198 139 141 108 112 113 121 376
```

```
round(exp,1)
```

```
333.5 192.9 137.7 123.2 120.7 117.3 116.2 136.7 355.7
```

```
#computing chi-squared statistic
print(chi.sq<- sum((obs-exp)^2/exp))
```

```
7.494979
```

```
#computing p-value
print(p.value<- 1-pchisq(chi.sq, df=3))
```

```
0.05768759
```

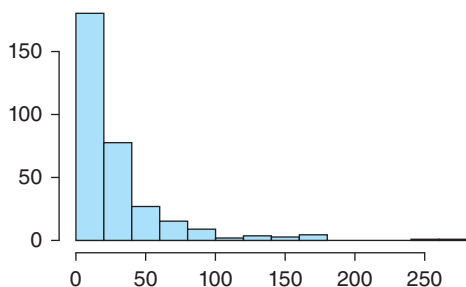
Note that we estimated five parameters in the polynomial regression, therefore, we needed to pick at least seven bins to have a non-degenerate number of degrees of freedom for the test. The total time span in the data set is 3,647 days (without the outlier), so it was reasonable to divide the range into 9 bins of size 400 days each (the last bin has 447 days). Nine bins result in 3 degrees of freedom, ($df = 9 - 1 - 5 = 3$).

Looking at the chi-squared statistic and the corresponding p -value, we can conclude at the 5% level of significance that the mean values in this process are well modeled by the fitted integrated intensity rate function.

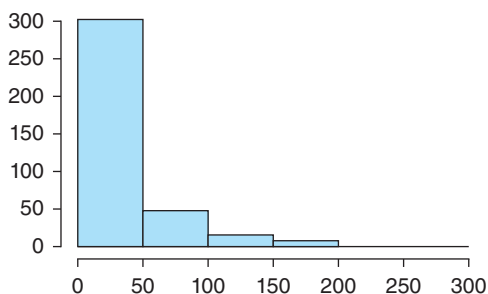
It remains to show that the interarrival times have an exponential distribution. It is not possible to do it rigorously because of the nonhomogeneous nature of the process, but at least we can construct histograms for the interarrival times in each of the nine bins to see that they have the shape of an exponential

density. As an illustration, below we present the histograms for the first and the last bins.

```
int1<- eq.data$elapsed.time[as.Date(eq.data$DATE)
<"2011/10/07"]
hist(int1, main="", xlab="", ylab="", col="light blue")
```



```
int9<- eq.data$elapsed.time[as.Date(eq.data$DATE)
>="2019/06/07"]
hist(int9, main="", xlab="", ylab="", col="light blue")
```



□

APPLICATION 4.2. Reliability engineers are concerned with the ability of manufactured systems or components to function without failure. Once failed, the item is repaired (in a repairable system) or replaced (in a non-repairable system). A stochastic model of the number of failures that has been widely used in practice by reliability engineers is a nonhomogeneous Poisson process with the *power-law intensity rate* (or *repair rate*) $\lambda(t) = \alpha\beta t^{\beta-1}$, $\alpha, \beta > 0$, $t \geq 0$. This function is very flexible because it models increasing rates if $\beta > 1$, or decreasing rates if $0 < \beta < 1$. If $\beta = 2$, the failure rate function degenerates into $\lambda(t) = 2\alpha t$, which corresponds to the homogeneous Poisson process.

(a) Let us study this model. Denote by $\{N(t), t \geq 0\}$ the number of failures by time t . The integrated intensity function is $\Lambda(t) = \int_0^t \alpha \beta u^{\beta-1} du = \alpha t^\beta$, $\alpha, \beta > 0, t \geq 0$. The probability mass function of $N(t)$ has the form

$$\mathbb{P}(N(t) - N(s) = n) = \frac{(\alpha t^\beta - \alpha s^\beta)^n}{n!} e^{-\alpha(t^\beta - s^\beta)}, \quad t \geq s \geq 0.$$

Note that the expected value of $N(t)$ is $\mathbb{E}(N(t)) = \Lambda(t) = \alpha t^\beta$.

An important question in reliability analysis is how to estimate α and β from the data. We address this question here. Two methods are commonly used to estimate the parameters. The first uses the linear regression approach, whereas the second one produces the maximum likelihood estimator.

METHOD 1 (LINEAR REGRESSION). Since $\mathbb{E}(N(t)) = \alpha t^\beta$, we can state the empirical analog $\hat{N}(t) = \hat{\alpha} t^{\hat{\beta}}$, or, equivalently, $\ln(\hat{N}(t)) = \ln \hat{\alpha} + \hat{\beta} \ln t$. Thus, $\ln(\hat{N}(t))$ can be regressed linearly on $\ln t$ to obtain the estimated intercept $\ln \hat{\alpha}$ and slope $\hat{\beta}$.

To look at a numeric example, assume that S_k , $k = 1, 2, \dots, 30$, the times to failure (in weeks) of certain auto parts during the pilot testing period, are as given in the table below. The second variable is $N(S_k) = k$, the total number of failures up to and including time S_k .

time	nfailures	time	nfailures	time	nfailures
2.36	1	10.16	11	19.70	21
2.96	2	10.87	12	20.99	22
4.71	3	11.32	13	21.56	23
5.23	4	13.36	14	22.57	24
6.16	5	14.52	15	22.79	25
7.15	6	16.19	16	24.02	26
7.33	7	16.84	17	25.8	27
8.20	8	17.19	18	26.49	28
8.45	9	18.18	19	27.13	29
9.31	10	18.72	20	28.05	30

Now we regress $\ln(N(S_k)) = \ln(k)$ on $\ln(S_k)$ to obtain $\hat{\alpha} = e^{-0.6854} = 0.5039$, and $\hat{\beta} = 1.2570$. The code and output are below.

```
reliability.data<- read.csv(file="./reliabilitydata.csv",
header=TRUE, sep=",")

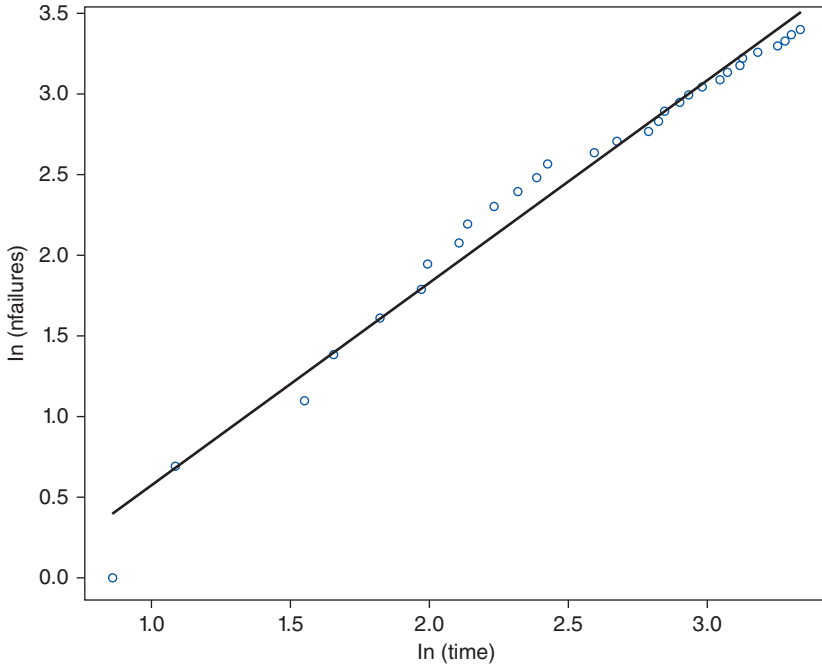
x<- log(reliability.data$time)
y<- log(reliability.data$nfailures)

glm(y~x)
```

Coefficients:

(Intercept)	x
-0.6854	1.2570

```
plot(x,y, xlab="ln(time)", ylab="ln(nfailures)")
lines(x, -0.6854+1.2570*x)
```



Using the known formulas for the estimators of the slope and intercept in linear regression, we write the explicit expressions for $\hat{\beta}$ and $\hat{\alpha}$. Here N is the total observed number of failures. In our example, $N = 30$. We have

$$\hat{\beta} = \frac{\sum_{k=1}^N [\ln(k) \ln(S_k)] - [\ln(N!)/N] \left[\sum_{k=1}^N \ln(S_k) \right]}{\sum_{k=1}^N [\ln(S_k)]^2 - (1/N) \left[\sum_{k=1}^N \ln(S_k) \right]^2},$$

and

$$\hat{\alpha} = \exp \left\{ (1/N) [\ln(N!) - \hat{\beta} \sum_{k=1}^N \ln(S_k)] \right\}.$$

We can verify in R that these expressions give us the same results as above. Indeed,

```
print(beta1.hat<- (sum(x*y)-log(factorial(N))*mean(x))/
(sum(x^2) -N*(mean(x)) ^2))
```

1.256956

```
print(alpha1.hat<- exp(log(factorial(N))/N-beta1.hat*mean(x)))
```

0.5038849

Before we go to the second method of estimation, we need to define the distribution of the failure times $S_k, k = 1, 2, \dots$. Given that the k th failure occurred at time s_k , the distribution of S_{k+1} is Weibull with the scale parameter α and the shape parameter β . The density is

$$f_{S_{k+1}|S_k}(t|s_k) = \alpha \beta t^{\beta-1} e^{-\alpha(t^\beta - s_k^\beta)}, \quad t \geq s_k,$$

and the cumulative distribution function is

$$F_{S_{k+1}|S_k}(t|s_k) = 1 - e^{-\alpha(t^\beta - s_k^\beta)}, \quad t \geq s_k.$$

METHOD 2 (MAXIMUM LIKELIHOOD ESTIMATOR). Suppose we observed N failures at times $S_1 = s_1, \dots, S_N = s_N$. We write the likelihood function as

$$\begin{aligned} L(\alpha, \beta | s_1, \dots, s_N) &= f_{S_N|S_{N-1}}(s_N | s_{N-1}) f_{S_{N-1}|S_{N-2}}(s_{N-1} | s_{N-2}) \cdot \dots \cdot \\ &\quad f_{S_2|S_1}(s_2 | s_1) f_{S_1}(s_1) \\ &= \alpha \beta s_N^{\beta-1} e^{-\alpha(s_N^\beta - s_{N-1}^\beta)} \cdot \alpha \beta s_{N-1}^{\beta-1} e^{-\alpha(s_{N-1}^\beta - s_{N-2}^\beta)} \cdot \dots \cdot \\ &\quad \alpha \beta s_2^{\beta-1} e^{-\alpha(s_2^\beta - s_1^\beta)} \cdot \alpha \beta s_1^{\beta-1} e^{-\alpha s_1^\beta} \\ &= (\alpha \beta)^N \left(\prod_{k=1}^N s_k \right)^{\beta-1} e^{-\alpha s_N^\beta}. \end{aligned}$$

Next, we find the expression for the log-likelihood function. We have

$$\ln(L) = N \ln \alpha + N \ln \beta + (\beta - 1) \sum_{k=1}^N \ln(s_k) - \alpha s_N^\beta.$$

Differentiating the log-likelihood function with respect to α and β and setting the expressions equal to 0, we obtain

$$\frac{\partial \ln(L)}{\partial \alpha} = \frac{N}{\alpha} - s_N^\beta = 0,$$

and

$$\frac{\partial \ln(L)}{\partial \beta} = \frac{N}{\beta} + \sum_{k=1}^N \ln(s_k) - \alpha s_N^\beta \ln(s_N) = 0.$$

From the first equation, $\alpha s_N^\beta = N$, and thus, the second equation can be rewritten as

$$\frac{N}{\beta} + \sum_{k=1}^N \ln(s_k) = \alpha s_N^\beta \ln(s_N) = N \ln(s_N).$$

From here,

$$\hat{\beta} = \frac{N}{N \ln(s_N) - \sum_{k=1}^N \ln(s_k)} = \left[(1/N) \sum_{k=1}^N \ln(s_N/s_k) \right]^{-1}, \text{ and } \hat{\alpha} = N/s_N^{\hat{\beta}}.$$

Going back to our numeric example, we write syntax in R, utilizing the variable `x` that contains the logs of failure times.

```
print(beta2.hat<- N/(N*x[N]-sum(x)))
```

```
1.236357
```

```
print(alpha2.hat<- N/exp(x[N]*beta2.hat))
```

```
0.4863609
```

(b) Another essential question in reliability analysis concerns the prediction of the next failure time. Suppose we have observed N failure times and the last one was at s_N . We can estimate S_{N+1} by its conditional mean value. We write

$$\mathbb{E}(S_{N+1} | S_N = s_N) = \int_{s_N}^{\infty} \alpha \beta t^\beta e^{-\alpha(t^\beta - s_N^\beta)} dt.$$

Now we use the substitution $u = \alpha t^\beta$, from where $\alpha \beta t^\beta dt = t du = \alpha^{-1/\beta} u^{1/\beta} du$. So, we obtain

$$\mathbb{E}(S_{N+1} | S_N = s_N) = \alpha^{-1/\beta} e^{\alpha s_N^\beta} \int_{\alpha s_N^\beta}^{\infty} u^{1/\beta} e^{-u} du.$$

The integral can be calculated in R as an upper incomplete gamma function $\int_x^\infty u^{a-1} e^{-u} du$ with the parameter $a = 1/\beta + 1$, and the lower limit of integration $x = \alpha s_N^\beta$. The following code computes the prediction. It uses the maximum likelihood estimators of α and β .

```
library(pracma)
alpha2.hat^(-1/beta2.hat)*exp(alpha2.hat*exp(x[N])
^beta2.hat)*
gammainc(alpha2.hat*exp(x[N])^beta2.hat, 1/beta2.hat+1)[2]
```

28.80161

Thus, given that the 30th failure was observed at 28.05 weeks, we predict that the 31st failure will occur at 28.8 weeks. \square

Exercises

EXERCISE 4.1. While still under the manufacturer's warranty, calculators break down during the first three years with the rate of 3 per year. Between three and ten years, the rate increases linearly from 3 per year to 17 per year.

- What stochastic model can be used to model the number of broken calculators? Specify all parameters.
- Find the probability that 50 calculators break down between year 4 and year 8.
- Find the average number of calculators that will break down between year 2 and year 10.

EXERCISE 4.2. Occurrence of wildfires in a certain area during a 120-day fire season can be modeled as a nonhomogeneous Poisson process with the intensity function $\lambda(t) = -0.000025 t^3 + 0.002 t^2 + 0.12 t$, where $0 \leq t \leq 120$.

- Plot the intensity function. Discuss its behavior. When is the peak of the intensity rate?
- Plot the integrated intensity function. Find the average number of wildfires per season.
- Find the average number of wildfires during the middle 50% of the season.

EXERCISE 4.3. Workers' injuries at an industrial manufacturing plant occur according to a nonhomogeneous Poisson process with the rate function $\lambda(t) = A/\sqrt{t}$, $t \geq 0$.

- (a) Given that 30 injuries happened, on average, during the first year of plant's operation, find the value of A .
- (b) Find the distribution of the times elapsing between two injuries. Simulate a trajectory of 100 injuries. What is the time range of the trajectory?
- (c) Assuming that the 100th injury occurred 12 years and 3 months after the plant was opened, simulate a trajectory.

EXERCISE 4.4. Road traffic or airport noise data are often modeled over time as a nonhomogeneous Poisson process. It is assumed that noise pollution above a certain threshold has an intensity rate that allows cyclic behavior of observations. For instance, suppose an environmental noise pollution has the intensity rate $\lambda(t) = 10 * (1 + \cos(2\pi t))$, $t \geq 0$. Use the thinning method to simulate a trajectory on the interval of length 10.

EXERCISE 4.5. In the process of radioactive decay, photons are emitted according to a nonhomogeneous Poisson process with the intensity rate $\lambda(t) = 100e^{-0.5t}$, $t \geq 0$. Use each of the three simulation methods to generate a trajectory. Fix parameters as 20 events for the first method, and the length of the time interval as 0.25 in the other two methods.

EXERCISE 4.6. The National Weather Service website contains the data on fatal lightning strikes in the United States. The file <https://www.weather.gov/media/hazstat/80years.pdf> gives the counts of yearly lightning fatalities between 1940 and 2019.

- (a) Plot the counts against year. Argue that the intensity rate function decays exponentially. Provide a possible explanation for this decay.
- (b) Details of US lightning deaths (in particular, the dates) are provided on the same website <https://www.weather.gov/safety/lightning-victims>. The data are given for 2006-2019. Use the data to support the statement that this natural phenomenon is not governed by a nonhomogeneous Poisson process. Hint: Are the interarrival times exponentially distributed?

EXERCISE 4.7. The capacity of cargo container ships and port terminals are traditionally measured in twenty-foot equivalent units (TEUs), the number of 20-foot-long containers. Suppose that containership arrival to a port can be modeled by a nonhomogeneous Poisson process with the power-law intensity function $\lambda(t) = \alpha\beta t^{\beta-1}$, $\alpha, \beta > 0$, $t \geq 0$. The data for 27 arrivals (in units of 10,000 TEUs) are provided in the table below.

Arrivals	Days	Arrivals	Days	Arrivals	Days
1	3.72	10	21.16	19	41.23
2	5.45	11	23.33	20	43.03
3	8.65	12	25.26	21	45.43
4	10.33	13	26.77	22	48.13
5	12.54	14	30.19	23	49.82
6	14.83	15	32.74	24	52.27
7	15.82	16	35.75	25	53.32
8	18.04	17	37.51	26	55.91
9	19.05	18	38.85	27	59.10

- (a) Estimate the parameters of the model using the regression approach.
- (b) Estimate the parameters of the model using the maximum likelihood approach.
- (c) Predict when the next 10,000 TEUs arrive at the port. Use both estimators from parts (a) and (b).

Compound Poisson Process

5.1 Definition of Compound Poisson Process

A stochastic process $\{X(t), t \geq 0\}$ is called a *compound Poisson process*¹ if

$X(t) = \sum_{i=1}^{N(t)} Y_i$ where $\{N(t), t \geq 0\}$ is a Poisson process with rate λ , and Y_i 's are independent and identically distributed random variables which are also independent of $\{N(t), t \geq 0\}$.

PROPOSITION 5.1. The mean and variance of a compound Poisson process are $\mathbb{E}(X(t)) = \lambda t \mathbb{E}(Y_1)$ and $\mathbb{V}ar(X(t)) = \lambda t \mathbb{E}(Y_1^2)$.

PROOF: The mean can be computed by conditioning on the value of $N(t)$. We write

$$\begin{aligned}\mathbb{E}(X(t)) &= \mathbb{E}[\mathbb{E}(X(t) | N(t))] = \mathbb{E}\left[\mathbb{E}\left(\sum_{i=1}^{N(t)} Y_i | N(t)\right)\right] \\ &= \mathbb{E}[N(t)\mathbb{E}(Y_1)] = \mathbb{E}(N(t)) \mathbb{E}(Y_1) = \lambda t \mathbb{E}(Y_1).\end{aligned}$$

Likewise, we compute the variance by conditioning on the value of $N(t)$. We get

$$\begin{aligned}\mathbb{V}ar(X(t)) &= \mathbb{E}[\mathbb{V}ar(X(t) | N(t))] + \mathbb{V}ar[\mathbb{E}(X(t) | N(t))] \\ &= \mathbb{E}\left[\mathbb{V}ar\left(\sum_{i=1}^{N(t)} Y_i | N(t)\right)\right] + \mathbb{V}ar\left[\mathbb{E}\left(\sum_{i=1}^{N(t)} Y_i | N(t)\right)\right] = \mathbb{E}(N(t)\mathbb{V}ar(Y_1)) \\ &\quad + \mathbb{V}ar(N(t)\mathbb{E}(Y_1)) = \mathbb{E}(N(t))\mathbb{V}ar(Y_1) + \mathbb{V}ar(N(t))(\mathbb{E}(Y_1))^2 = \lambda t \mathbb{V}ar(Y_1) \\ &\quad + \lambda t (\mathbb{E}(Y_1))^2 = \lambda t (\mathbb{E}(Y_1^2) - (\mathbb{E}(Y_1))^2) + \lambda t (\mathbb{E}(Y_1))^2 = \lambda t \mathbb{E}(Y_1^2). \quad \square\end{aligned}$$

¹The first treatment of the compound Poisson process is attributed to Filip Lundberg, a pioneer of the actuarial collective risk theory. In 1903, he wrote his Ph.D. dissertation at the University of Uppsala titled "Approximations of the Probability Function/Reinsurance of Collective Risks" (in Swedish).

EXAMPLE 5.1. Visitors walk into a casino in Las Vegas according to a Poisson process with a rate of 50 per hour. Ten percent of them will not gamble at all, others will lose independently a random number of dollars which we assume has a Uniform(0, \$1,500) distribution. We need to model the casino's gain.

To this end, we focus only on the gamblers who are entering the casino. Their arrival can be modeled by a Poisson process $\{N(t), t \geq 0\}$ with rate $(0.9)(50) = 45$ per hour. Let Y_i denote the amount each gambler loses. We are given that Y_i 's are independent and uniformly distributed with mean $\$1,500/2 = \750 and variance $(\$1,500)^2/12$. Consider $X(t) = \sum_{i=1}^{N(t)} Y_i$, the total sum of money that the gamblers lose at the casino within t hours. It is driven by a compound Poisson process.

(a) The casino's expected gain during a 12-hour period can be computed as $\mathbb{E}(X(12)) = (45)(12)(\$750) = \$405,000.00$.

(b) The standard deviation of the gain is

$$\sqrt{\text{Var}(X(12))} = \sqrt{(45)(12)((\$1,500)^2/12 + (\$750)^2)} = \$20,124.61. \quad \square$$

EXAMPLE 5.2. Suppose the number of car accidents at a certain intersection can be modeled by a Poisson process with rate $\lambda = 3$ per month. Assume that the number of people who are involved in each accident is a binomially distributed random variable with parameters $n = 8$ and $p = 0.3$. Then the total number of people involved in car accidents on that intersection within a time period of t months can be modeled by a compound Poisson process $\{X(t) = \sum_{i=1}^{N(t)} Y_i, t \geq 0\}$ where $N(t) \sim \text{Poisson}(3t)$, and $Y_i, i = 1, 2, \dots, N(t)$, are independent random variables with $\text{Bi}(8, 0.3)$ distribution, also independent of $N(t)$. The average number of people involved in car accidents on this intersection within one year is $\mathbb{E}(X(12)) = \lambda t np = (3)(12)(8)(0.3) = 86.4$ with the standard deviation

$$\begin{aligned} \sqrt{\text{Var}(X(12))} &= \sqrt{\lambda t \mathbb{E}(Y_1^2)} = \sqrt{\lambda t (\text{Var}(Y_1) + (\mathbb{E}(Y_1))^2)} \\ &= \sqrt{\lambda t (np(1-p) + n^2 p^2)} = \sqrt{(3)(12)((8)(0.3)(0.7) + (8)^2(0.3)^2)} = 16.36582. \end{aligned}$$

□

EXAMPLE 5.3. Families enter a movie theater according to a Poisson process with rate $\tilde{\lambda} = 15$ per hour. The number of family members is distributed according to a zero-truncated Poisson distribution with rate $\lambda = 3$. We are interested in modeling the total number of moviegoers who enter the movie theater by time t .

Denote by $\{N(t), t \geq 0\}$ the Poisson process that describes family arrivals, and let Y_i , $i = 1, 2, \dots, N(t)$, be the size of the i th family entering. We are given that $N(t) \sim \text{Poisson}(15t)$ and is independent of Y_i 's, which, in turn, are independent and identically distributed with the probability mass function $\mathbb{P}(Y_i = n) = \frac{\lambda^n}{n!} \frac{e^{-\lambda}}{1 - e^{-\lambda}}$, $n = 1, 2, \dots$. The total number of movie goers can be described by a compound Poisson process $\{X(t) = \sum_{i=1}^{N(t)} Y_i, t \geq 0\}$. To find the mean and variance of this process, we first derive the expressions for the mean and second moment of a zero-truncated Poisson distribution. We write

$$\mathbb{E}(Y_1) = \frac{1}{e^\lambda - 1} \sum_{n=1}^{\infty} \frac{n\lambda^n}{n!} = \frac{\lambda}{e^\lambda - 1} \sum_{n=1}^{\infty} \frac{\lambda^{n-1}}{(n-1)!} = \frac{\lambda e^\lambda}{e^\lambda - 1},$$

and

$$\begin{aligned} \mathbb{E}(Y_1^2) &= \frac{1}{e^\lambda - 1} \sum_{n=1}^{\infty} \frac{n^2 \lambda^n}{n!} = \frac{1}{e^\lambda - 1} \left[\sum_{n=1}^{\infty} \frac{n(n-1)\lambda^n}{n!} + \sum_{n=1}^{\infty} \frac{n\lambda^n}{n!} \right] \\ &= \frac{1}{e^\lambda - 1} [\lambda^2 e^\lambda + \lambda e^\lambda] = \frac{\lambda(\lambda+1)e^\lambda}{e^\lambda - 1}. \end{aligned}$$

Thus, the average number of people who enter the movie theater during a t -hour period is

$$\mathbb{E}(X(t)) = \tilde{\lambda} t \mathbb{E}(Y_1) = \frac{\tilde{\lambda} t \lambda e^\lambda}{e^\lambda - 1},$$

with the standard deviation

$$\sqrt{\text{Var}(X(t))} = \sqrt{\tilde{\lambda} t \mathbb{E}(Y_1^2)} = \sqrt{\frac{\tilde{\lambda} t \lambda (\lambda+1) e^\lambda}{e^\lambda - 1}}.$$

For instance, during a 6-hour period, the movie theater can expect $\mathbb{E}(X(6)) = \frac{(15)(6)(3)e^3}{e^3 - 1} = 284.1468$ visitors, with a standard deviation of $\sqrt{\text{Var}(X(6))} = \sqrt{\frac{(15)(6)(3)(3+1)e^3}{e^3 - 1}} = 33.71331$ visitors. \square

5.2 Simulations in R

In this section, we use the setting of Example 5.1 and simulate trajectories of the casino's gain. First, we generate a Poisson process of gambler's arrivals and then generate the amounts lost by the gamblers, which are independent random variables uniformly distributed on $(0, \$1,500)$. The sum of the loss

amounts up to time t is the desired compound Poisson process. We generate the Poisson arrivals by the two methods described in [Chapter 3](#), by fixing the number of arrivals at 20 gamblers and generating independent exponentially distributed interarrival times, and by fixing the time interval at 20 minutes and generating event times as the order statistics from the uniform distribution on $[0, 20]$. The codes and graphs are given below.

SIMULATION 5.1. (EXPONENTIAL INTERARRIVALS).

```
#specifying parameters
lambda<- 0.75
narrivals<- 20

#defining casino gain and time as vectors gain<- c()
time<- c()

#setting initial values
gain[1]<- 0
time[1]<- 0

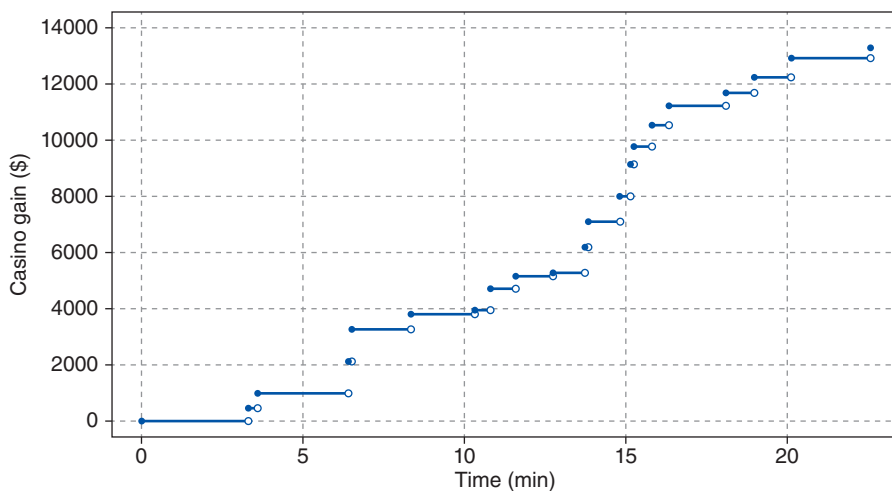
#specifying seed
set.seed(50094)

#simulating trajectory
for (i in 2:(narrivals+1)) {
  time[i]<- time[i-1] - 1/lambda*log(runif(1))
  gain[i]<- gain[i-1] + runif(1,0, 1500)
}

#plotting trajectory
plot(time, gain, type="n", ylim=c(0,14000), xlab="Time
(min)", ylab="Casino gain ($)", panel.first = grid())

segments(time[-length(time)], gain[-length(time)],
time[-1]-0.15, gain[-length(time)], lwd=2, col="blue")

points(time, gain, pch=20, col="blue"))
points(time[-1], gain[-length(time)], pch=1, col="blue"))
```



```
time[length(time)]
```

22.561

```
gain[length(gain)]
```

13291.65

In this simulated trajectory, 20 gamblers walked into the casino (a pre-determined number of arrivals). They all came within 22.561 minutes and lost cumulatively \$13,291.65. \square

SIMULATION 5.2. (UNIFORM ORDER STATISTICS).

```
#specifying parameters
t<- 20
lambda<- 0.75

#specifying seed
set.seed(41130)

#generating number of arrivals
narrivals<- rpois(1,lambda*t)

#defining vectors
gain<- c()
time<- c()
u<- c()
```



```

#setting initial values
gain[1]<- 0
u[1]<- 0

#generating standard uniforms and gain
for(i in 2:(narrivals+1)) {
  u[i]<- runif(1)
  gain[i]<- gain[i-1] + runif(1,0, 1500)
}

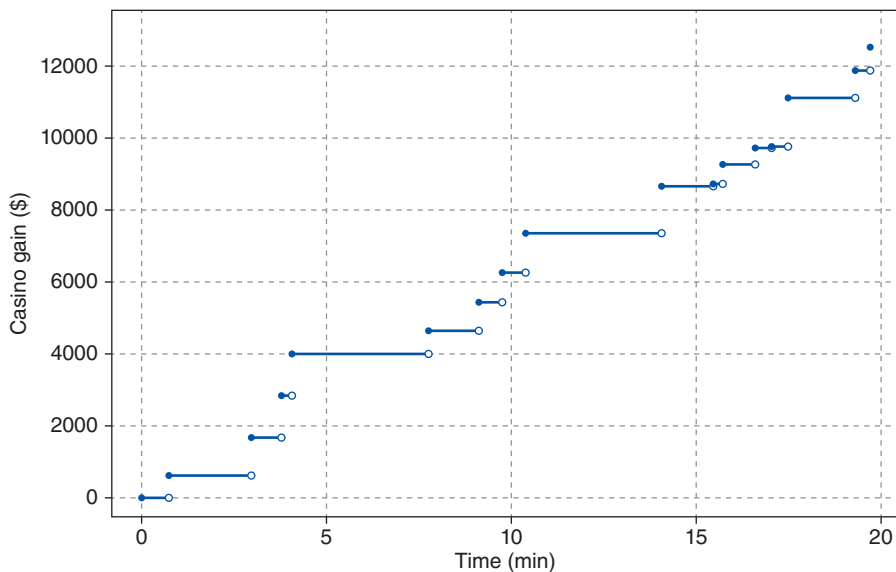
#computing event times
time<- t*sort(u)

#plotting trajectory
plot(time, gain, type="n", ylim=c(0,13000), xlab="Time
(min)", ylab="Casino gain ($)", panel.first = grid())

segments(time[-length(time)], gain[-length(time)],
time[-1]-0.15, gain[-length(time)], lwd=2, col="blue")

points(time, gain, pch=20, col="blue")
points(time[-1], gain[-length(time)], pch=1, col="blue")

```



```
narrivals
```

16

```
gain[length(gain)]
```

12504.86

In this simulated trajectory, within a fixed time period of 20 minutes, 16 gamblers walked into the casino and lost a total of \$12,504.86. \square

5.3 Applications of Compound Poisson Process

APPLICATION 5.1. A compound Poisson *collective risk model* is a classical model in the actuarial field. It assumes that claims are submitted according to a Poisson distribution with rate λ , and that the amount of claims have a certain known distribution. Then the *aggregate claim amount* up to time t

is a compound Poisson process $\{X(t) = \sum_{i=1}^{N(t)} Y_i, t \geq 0\}$ where $N(t)$ denotes

the number (or *frequency*) of claims, and Y_i is the amount (or *severity*) of the i th claim. As in any compound Poisson model, $N(t)$ and all Y_i 's are assumed independent. We know that $\mathbb{E}(X(t)) = \lambda t \mathbb{E}(Y_1)$ and $\text{Var}(X(t)) = \lambda t \mathbb{E}(Y_1^2)$.

The main question that actuaries have to answer is how much money should be collected in premiums so that the company will be able to pay the claims. Let $L(t) = X(t) - ct$ be the insurer's loss. It represents the difference between the total benefit payments that the company has to make and the amount of premiums collected over time with a constant rate c . It is customary to consider c of the form $c = (1 + \theta) \lambda \mathbb{E}(Y_1)$ where θ is termed a *security loading*. It means that up to time t , the company will collect in premiums the amount $ct = (1 + \theta) \lambda t \mathbb{E}(Y_1) = (1 + \theta) \mathbb{E}(X(t))$, which gives the company some cushion above the expected aggregate claim amount $\mathbb{E}(X(t))$ in case there are some unusually high claims. One of several ways to find the value of the security loading θ is to assume that the company wants to see a positive loss at most, say, 5% of the time. Thus, θ solves $\mathbb{P}(L(t) > 0) = 0.05$ or $\mathbb{P}(X(t) - (1 + \theta) \mathbb{E}(X(t)) > 0) = 0.05$. We can rewrite this identity as

$$\mathbb{P}\left(\frac{X(t) - \mathbb{E}(X(t))}{\sqrt{\text{Var}(X(t))}} > \theta \frac{\lambda t \mathbb{E}(Y_1)}{\sqrt{\lambda t \mathbb{E}(Y_1^2)}}\right) = 0.05.$$

Now, assuming that λ is large, we can use the Central Limit Theorem to conclude that $\frac{X(t) - \mathbb{E}(X(t))}{\sqrt{\text{Var}(X(t))}}$ has approximately a standard normal distribution. Hence, θ can be found as the solution of the equation

$$\mathbb{P}\left(Z > \theta \frac{\sqrt{\lambda t} \mathbb{E}(Y_1)}{\sqrt{\mathbb{E}(Y_1^2)}}\right) = 0.05.$$

That is,

$$\theta = 1.645 \frac{\sqrt{\mathbb{E}(Y_1^2)}}{\sqrt{\lambda t} \mathbb{E}(Y_1)}.$$

As an illustration, we consider data on storms downloaded from the National Oceanic and Atmospheric Administration's site (<https://www.ncdc.noaa.gov/stormevents/>). The data set contains dates, times, and amounts of damage (in units of \$1,000) in all counties in Texas from March to April of 2020. The reported damage was done during a storm by hail, wind, tornado, flash flood, or lightning. There are a total of 85 rows in this data set. Damages range between \$500 and \$150,000, with two additional values of \$500,000 and \$800,000. Assuming that a single insurance company took care of all the claims, below we evaluate the security loading that this company must utilize when calculating premiums.

```
storm.data<- read.csv(file="./stormdata.csv", header=TRUE,
  sep=",")
```

```
#creating date-time variable
storm.data$datetime<- as.POSIXct
(paste(as.Date(storm.data$Date), storm.data$Time))
```

```
#estimating event rate
print(nevents<- nrow(storm.data))
```

85

```
print(ndays<- (as.numeric(storm.data$datetime[nevents])
-as.numeric(storm.data$datetime[1]))/(24*3600))
```

46.09028

```
print(lambda.hat<- nevents/ndays)
```

1.844207

Within 46.09 days, there were 85 storms with tangible damage (resulting in the insurance claims). It means that the claims were submitted with a rate of 1.844207 per day. Finally, we estimate θ , using the empirical values of the first and second moments of the damage amounts.

```
#estimating security loading
print(theta<- 1.645*sqrt(mean(storm.data$Damage^2))/
      (sqrt(lambda.hat*ndays)*mean(storm.data$Damage)))
```

0.5516492

It means that the company should collect about 155% of the mean claim amount to hedge against large claims. \square

Exercises

EXERCISE 5.1. The producer of a television game show with cash prizes wants to set a budget equal to the expected value plus one standard deviation of the aggregate cash prizes. The number of cash prizes given is a Poisson process with a rate of 1.5 per hour. Each episode lasts for 2 hours. The distribution of prize amounts is as follows:

Prize Amount	\$5,000	\$2,000	\$500	\$100
Probability	0.15	0.35	0.2	0.3

- Calculate the budget for 100 episodes of the game show.
- Simulate a trajectory of 100 games. For the trajectory, what is the amount of the budget left after the 100th game? If the amount is negative, during or after which episode did the producer run out of money?

EXERCISE 5.2. When a pharmacy bills the medical insurance company, the claims arrive as a Poisson process with the rate $\lambda = 60$ per day. The amounts of claims are independent and uniformly distributed between \$30 and \$300. It is also assumed that the amounts of the claims and the number of claims are independent.

- What is the expected aggregate claim amount that the medical insurance company receives within a 30-day period? What is the standard deviation of this amount?
- Use the Central Limit Theorem to approximate the probability that the aggregate claim amount will exceed \$300,000 within a 30-day period.

EXERCISE 5.3. The photon detection process in X-ray computed tomography can be modeled as a compound Poisson process. The X-ray photons collide with a photon detector and then generate some number of light photons. The number of incident X-ray photons changes according to a Poisson process with

a rate of λ per second. The number of light photons generated by each X-ray photon that is detected is a Poisson random variable with a rate of $\hat{\lambda}$ per second.

- (a) Define the aggregate number of light photons that are generated up to t seconds. Give the formula for the process, and the expressions for its mean and standard deviation.
- (b) Assuming that the average rate of X-ray photons is 50 per second and the mean of light photons is 5, simulate 100 values of the aggregate number of light photons generated within a 10-second period.
- (c) Construct a histogram for the 100 values generated in part (b). Is the histogram approximately bell-shaped? Explain.

EXERCISE 5.4. An insurance company receives claims according to a Poisson process $\{N(t), t \geq 0\}$ with rate λ . Assume that claim amounts $X_i, i = 1, 2, \dots$, are independent and identically distributed, and are independent of claim arrival times $S_i, i = 1, 2, \dots$. The present-day (day of policy issue) value of the amount of claim X_i made at time S_i is computed as $X_i e^{-\delta S_i}$, where δ is the force of interest. The present-day value of the total claim amount $P(t) = \sum_{i=1}^{N(t)} X_i e^{-\delta S_i}$ changes according to a compound Poisson process. Show that the mean of this process is $\mathbb{E}[P(t)] = \mathbb{E}(X_1)(\lambda/\delta)(1 - e^{-\delta t})$.

EXERCISE 5.5. In radiobiology, when cells are exposed to radiation, DNA sometimes breaks, and the broken ends may abnormally rejoin resulting in chromosome aberrations. The number of ions that traverse through a cell nucleus by time t is modeled as a Poisson process with rate λ . Each traversal independently causes Y_i aberrations which have Poisson distribution with rate β . Let $X(t) = \sum_{i=1}^{N(t)} Y_i$ denote the total number of chromosomal aberrations by time t .

- (a) Show that the compound Poisson process $X(t)$ has the Neyman type A distribution with the probability function given by

$$\mathbb{P}(X(t) = x) = \frac{\beta^x}{x!} e^{-\lambda t} \sum_{n=0}^{\infty} \frac{n^x (\lambda t)^n}{n!} e^{-\beta n}.$$

- (b) Show that the mean of $X(t)$ is $\lambda t \beta$ and the variance is $\lambda t (\beta + \beta^2)$.
- (c) For a fixed t , denote the sample mean of $X(t)$ by $\hat{E}(X(t))$, sample variance by $\hat{Var}(X(t))$, and empirical probability of zero $\mathbb{P}(X(t) = 0)$ by $\hat{P}(0)$. Show that β and λ may be estimated respectively by $\hat{\beta} = \frac{\hat{Var}(X(t))}{\hat{E}(X(t))} - 1$ and

$$\hat{\lambda} = \frac{-\ln(\hat{P}(0))}{t(1 - e^{-\hat{\beta}})}.$$

EXERCISE 5.6. Cars arrive at a gas station according to a Poisson process. Each car driver buys gas independently of others for a dollar amount that has a gamma distribution. The spent amounts and the number of cars are independent. Data for times of car arrivals (in minutes) and dollar amounts spent are presented in the table below.

Arrival Time	Amount Spent, in \$	Arrival Time	Amount Spent, in \$	Arrival Time	Amount Spent, in \$
0.15	23.67	28.81	69.67	47.94	20.38
3.81	25.55	32.36	25.39	49.73	34.95
5.67	38.54	32.76	30.86	50.72	29.23
6.61	31.31	32.92	50.53	50.86	36.51
13.14	74.20	33.22	24.93	51.99	37.77
13.57	32.78	33.51	27.49	52.36	34.41
15.68	29.70	34.40	22.56	52.89	23.35
22.83	35.83	35.76	21.38	53.64	32.95
23.35	22.17	39.08	45.53	55.03	21.27
23.77	34.96	41.03	39.14	55.29	37.32
23.77	24.20	42.05	26.02	56.82	20.30
24.69	26.01	42.38	21.35	63.02	32.59
26.94	24.07	45.66	33.88		

(a) Argue that the total dollar amount that the gas station receives can be modeled by a compound Poisson process. Write down the expression for the process and describe all parameters.

(b) Estimate the parameters of the model using the method of moments. Show that the estimators of α and β of the gamma distribution are $\hat{\alpha} =$

$$\frac{n\bar{Y}^2}{\sum_{i=1}^n Y_i^2 - n\bar{Y}^2} \text{ and } \hat{\beta} = \frac{\bar{Y}}{\hat{\alpha}}.$$

(c) Plot histograms for the interarrival times and the amount spent, with fitted distribution curves.

(d) Write down the estimates of the mean and standard deviation of the total dollar amount at 1 hour. Use the parameter estimates obtained in part (b).



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Conditional Poisson Process

6.1 Definition of Conditional Poisson Process

A counting process $\{N(t), t \geq 0\}$ is called a *conditional* (or *mixed*) Poisson process¹ if $N(t)$ has a Poisson distribution with rate Λ where Λ is a random variable with a known distribution $f_\Lambda(\lambda)$. The rate Λ is referred to as a *random intensity rate*.

The probability mass function for the conditional Poisson process is specified as a conditional probability

$$\mathbb{P}(N(t+s) - N(s) = n \mid \Lambda = \lambda) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}, \quad t, s \geq 0, n = 0, 1, 2, \dots$$

The marginal distribution of $N(t)$ is found as

$$\mathbb{P}(N(t+s) - N(s) = n) = \int_0^\infty \frac{(\lambda t)^n}{n!} e^{-\lambda t} f_\Lambda(\lambda) d\lambda, \quad t, s \geq 0, n = 0, 1, 2, \dots$$

REMARK 6.1. A conditional Poisson process has stationary increments. It is reflected in the above formulas for conditional and marginal distributions of $N(t)$. The above expressions do not depend on s , the beginning of the interval, only on t , the length of the interval. The increments are not independent, though. It is easily seen if we write

$$\begin{aligned} \mathbb{P}(N(s) = n, N(t-s) = m) &= \int_0^\infty \mathbb{P}(N(s) = n, N(t-s) = m \mid \Lambda = \lambda) f_\Lambda(\lambda) d\lambda \\ &= \int_0^\infty \mathbb{P}(N(s) = n \mid \Lambda = \lambda) \mathbb{P}(N(t-s) = m \mid \Lambda = \lambda) f_\Lambda(\lambda) d\lambda \end{aligned}$$

¹Introduced in Dubourdieu, J. (1938). “Remarques relatives a la théorie mathématique de l’assurance-accidents.” *Bulletin Trimestriel de l’Institut des Actuaire Français*, 49: 76 – 126.

$$\begin{aligned}
& \neq \int_0^\infty \mathbb{P}(N(s) = n \mid \Lambda = \lambda) f_\Lambda(\lambda) d\lambda \int_0^\infty \mathbb{P}(N(t-s) = m \mid \Lambda = \lambda) f_\Lambda(\lambda) d\lambda \\
& = \mathbb{P}(N(s) = n) \mathbb{P}(N(t-s) = m). \quad \square
\end{aligned}$$

PROPOSITION 6.1. The mean of $N(t)$ is $\mathbb{E}(N(t)) = t \mathbb{E}(\Lambda)$, and the variance is $\mathbb{V}ar(N(t)) = t^2 \mathbb{V}ar(\Lambda) + t \mathbb{E}(\Lambda)$.

PROOF: Conditioning on Λ , we write

$$\mathbb{E}(N(t)) = \mathbb{E}[\mathbb{E}(N(t) \mid \Lambda)] = \mathbb{E}(\Lambda t) = t \mathbb{E}(\Lambda),$$

and

$$\begin{aligned}
\mathbb{V}ar(N(t)) &= \mathbb{V}ar[\mathbb{E}(N(t) \mid \Lambda)] + \mathbb{E}[\mathbb{V}ar(N(t) \mid \Lambda)] \\
&= \mathbb{V}ar[\Lambda t] + \mathbb{E}[\Lambda t] = t^2 \mathbb{V}ar(\Lambda) + t \mathbb{E}(\Lambda). \quad \square
\end{aligned}$$

EXAMPLE 6.1. Assume that Λ in a conditional Poisson process has the probability mass function

$$\mathbb{P}(\Lambda = \lambda_0) = p_0 \text{ and } \mathbb{P}(\Lambda = \lambda_1) = 1 - p_0.$$

The marginal distribution of $N(t)$ is

$$\mathbb{P}(N(t) = n) = \frac{(\lambda_0 t)^n}{n!} e^{-\lambda_0 t} p_0 + \frac{(\lambda_1 t)^n}{n!} e^{-\lambda_1 t} (1 - p_0), n = 0, 1, 2, \dots$$

This is a mixture of two Poisson processes with rates λ_0 and λ_1 and the mixture weights p_0 and $1 - p_0$. As a mixture of two Poisson processes, $N(t)$ has mean $\mathbb{E}(N(t)) = \lambda_0 t p_0 + \lambda_1 t (1 - p_0) = t(\lambda_0 p_0 + \lambda_1 (1 - p_0)) = t \mathbb{E}(\Lambda)$, and variance

$$\begin{aligned}
\mathbb{V}ar(N(t)) &= p_0(\lambda_0 t + (\lambda_0 t)^2) + (1 - p_0)(\lambda_1 t + (\lambda_1 t)^2) - (\lambda_0 t p_0 + \lambda_1 t (1 - p_0))^2 \\
&= t^2 (\lambda_0^2 p_0 + \lambda_1^2 (1 - p_0) - (\lambda_0 p_0 + \lambda_1 (1 - p_0))^2) + t (\lambda_0 p_0 + \lambda_1 (1 - p_0)) \\
&= t^2 \mathbb{V}ar(\Lambda) + t \mathbb{E}(\Lambda). \quad \square
\end{aligned}$$

EXAMPLE 6.2. Suppose that in a conditional Poisson process, Λ has an exponential distribution with mean λ . The marginal probability mass function of $N(t)$ is

$$\begin{aligned}
\mathbb{P}(N(t) = n) &= \int_0^\infty \frac{(u t)^n}{n!} e^{-ut} \frac{1}{\lambda} e^{-u/\lambda} du \\
&= \frac{t^n}{(t + 1/\lambda)^{n+1} \lambda} \int_0^\infty \frac{u^n (t + 1/\lambda)^{n+1}}{n!} e^{-(t+1/\lambda)u} du \\
&= \frac{t^n}{(t + 1/\lambda)^{n+1} \lambda} = \frac{1}{\lambda t + 1} \left(1 - \frac{1}{\lambda t + 1}\right)^n, \quad n = 0, 1, 2, \dots
\end{aligned}$$

Therefore, the marginal distribution of $N(t)$ is geometric. It represents the number of failures until the first success. The success probability is $p = 1/(\lambda t + 1)$. It is known that the mean is $\frac{1-p}{p} = \frac{1-1/(\lambda t + 1)}{1/(\lambda t + 1)} = \lambda t = t \mathbb{E}(\Lambda)$, and variance is $\frac{1-p}{p^2} = \frac{1-1/(\lambda t + 1)}{1/(\lambda t + 1)^2} = \lambda t(\lambda t + 1) = t^2 \lambda^2 + t \lambda = t^2 \text{Var}(\Lambda) + t \mathbb{E}(\Lambda)$. \square

6.2 Simulations in R

SIMULATION 6.1. (EXPONENTIAL INTERARRIVALS). Consider the setting of Example 6.1. Suppose $p_0 = 0.3$, $\lambda_0 = 4$, and $\lambda_1 = 0.5$. Below we simulate 5 trajectories with 20 Poisson occurrences each, by generating exponential interarrival times. We first choose the rate randomly from the given binary distribution. Three of the trajectories have a rate of 4 and the other two have a rate of 0.5.

```
#specifying parameters
p<- c(0.3, 0.7)
lambda<- c(4, 0.5)
njumps<- 20

#specifying states and times as data frames
time<- data.frame()
N<- data.frame()

#specifying seed
set.seed(6335044)

#simulating trajectories
for(j in 1:5) {

#fixing the value for rate
Lambda<- lambda[sample(1:2, 1, prob=p)]

#setting initial values
time[1,j]<- 0
N[1,j]<- 0

#simulating trajectory
i<- 2
```

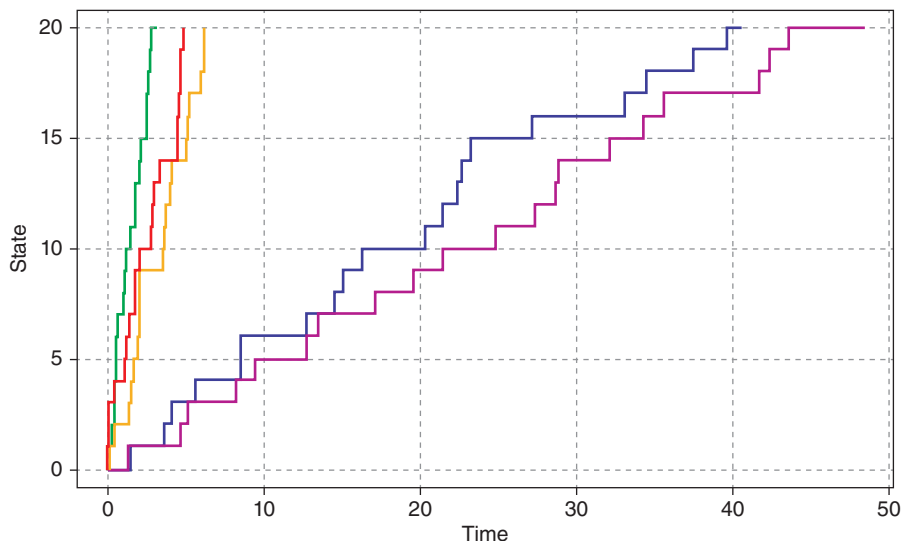
```

repeat {
time[i,j]<- time[i-1,j]+round((-1/Lambda)*log(1-runif(1)),3)
-0.001
N[i,j]<- N[i-1,j]

if(i==2*njumps+2) break
else {
time[i+1,j]<- time[i,j]+0.001
  N[i+1,j]<- N[i,j]+1
  i<- i+2
}
}
}

#plotting trajectories
matplot(time, N, type="l", lty=1, lwd=2, col=c("blue",
"green", "red", "purple", "orange"), xlab="Time",
ylab="State", panel.first=grid())

```



□

SIMULATION 6.2. (UNIFORM ORDER STATISTICS). Now we use the uniform order statistics method described in earlier chapters to simulate five trajectories. We fix the end-time (say, $t = 30$), and then pick the rate λ from the

binary distribution. After that, we randomly choose the number of events from the Poisson (λt) distribution. Note that we expect trajectories with $\lambda = 0.5$ to have about $(30)(0.5) = 15$ occurrences, whereas trajectories with $\lambda = 4$ will have about $(30)(4) = 120$ occurrences. Below is the code and the graph. Two trajectories happened to have a rate of 4 and the other three have a rate of 0.5.

```
#specifying parameters
t<- 30
p<- c(0.3, 0.7)
lambda<- c(4, 0.5)

#specifying states and times as data frames
time<- data.frame()
N<- data.frame()

#specifying seed
set.seed(1902238)

#simulating trajectories
for(j in 1:5) {

#fixing the value for rate
Lambda<- lambda[sample(1:2, 1, prob=p)]

#setting initial values
time[1,j]<- 0
N[1,j]<- 0

#generating total number of jumps
njumps<- rpois(1,Lambda*t)

#generating standard uniforms
u<- c()
u[1]<- 0
for(i in 2:njumps)
u[i]<- runif(1)

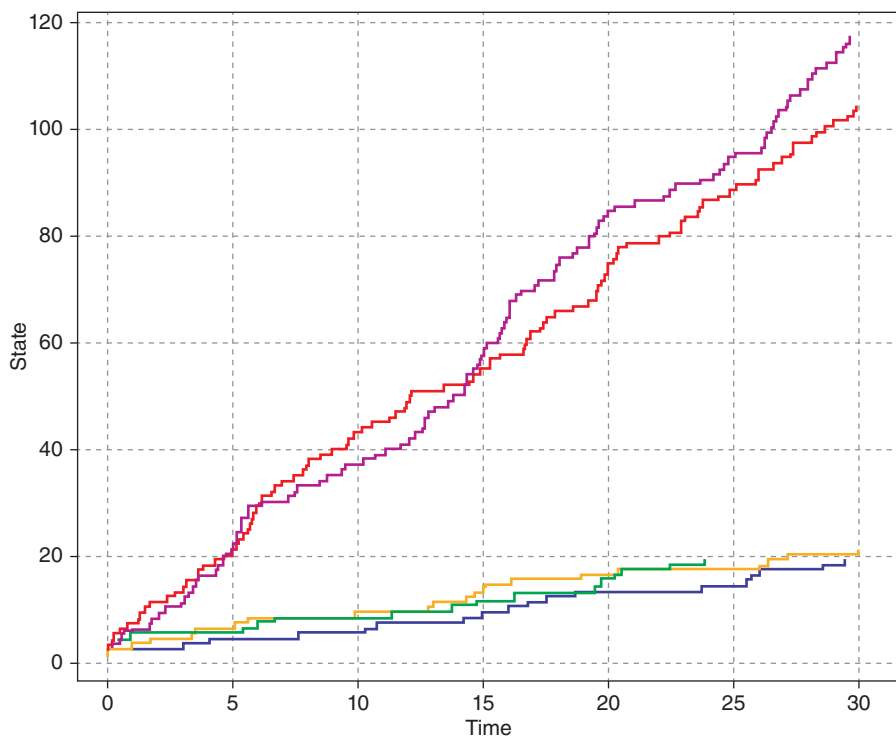
#computing event times
s<- t*sort(u)
```

```

#generating jumps
for (i in seq(2, 2*njumps, 2)) {
  time[i,j]<- s[i/2]-0.001
  time[i+1,j]<- s[i/2]
  N[i,j]<- N[i-1,j]
  N[i+1,j]<- N[i-1,j]+1
}
}

#plotting trajectories
matplot(time, N, type="l", lty=1, lwd=2, col=c("blue",
"green", "red", "purple", "orange"), xlab="Time",
ylab="State", panel.first=grid())

```



□

6.3 Applications of Conditional Poisson Process

APPLICATION 6.1. Agronomists model the presence of Colorado potato beetles as a conditional Poisson process. Suppose a plot of fertile soil is planted with potatoes. The width of the plot is fixed, so the area is proportional to the length. The number of egg clusters Λ that are located within a stretch of the plot of a certain length has a Poisson distribution with mean λ_0 . Each cluster contains between 10 and 30 eggs, but only several of them hatch. Suppose the number of eggs within a cluster that hatch follows a Poisson distribution with a rate r . Combining the results, we obtain that the total number of hatched eggs $N(\ell)$ on a stretch of the plot of length ℓ has a Poisson distribution with the rate $r\Lambda$, where $\Lambda \sim \text{Poisson}(\lambda_0)$.

(a) The expected value of $N(\ell)$ is computed by conditioning on Λ . We write $\mathbb{E}(N(\ell)) = \mathbb{E}[\mathbb{E}(N(\ell) | \Lambda)] = \mathbb{E}(r\ell\Lambda) = r\lambda_0\ell$. Likewise, the variance is computed by conditioning on Λ . We have $\mathbb{V}ar(N(\ell)) = \mathbb{V}ar[\mathbb{E}(N(\ell) | \Lambda)] + \mathbb{E}[\mathbb{V}ar(N(\ell) | \Lambda)] = \mathbb{V}ar(r\ell\Lambda) + \mathbb{E}(r\ell\Lambda) = (r\ell)^2\lambda_0 + (r\ell)\lambda_0 = \lambda_0(r\ell)(r\ell + 1)$.

To illustrate these with a numeric example, assume $r = 5$, $\ell = 2$, and $\lambda_0 = 3$. We evaluate $\mathbb{E}(N(\ell)) = (5)(3)(2) = 30$ and $\mathbb{V}ar(N(\ell)) = (3)(5)(2)((5)(2) + 1) = 330$.

(b) The marginal distribution of $N(\ell)$ is

$$\begin{aligned}\mathbb{P}(N(\ell) = n) &= \mathbb{E}[\mathbb{P}(N(\ell) = n | \Lambda)] = \mathbb{E}\left[\frac{(r\ell\Lambda)^n}{n!} e^{-r\ell\Lambda}\right] \\ &= \frac{(r\ell)^n}{n!} \mathbb{E}\left[\Lambda^n e^{-r\ell\Lambda}\right] = \frac{(r\ell)^n}{n!} M_{\Lambda}^{(n)}(-r\ell), \quad n = 0, 1, 2, \dots,\end{aligned}$$

where $M_{\Lambda}^{(n)}(-r\ell)$ is the n th derivative of the moment generating function of Λ , $M_{\Lambda}(t) = \exp\{\lambda_0(e^t - 1)\}$, computed at $t = -r\ell$. These derivatives have to be computed numerically. Below is the R code that calculates probabilities for $n = 0, \dots, 10$.

```
#specifying parameters
r<- 5
l<- 2
t<- -r*l
lambda0<- 3
```

```
#computing probabilities
prob<- c()
M<- expression(exp(lambda0*(exp(t)-1)))
prob[1]<- eval(M)

for (n in 2:11) {
  M<- D(M,"t")
  prob[n]<- (r*1)^n/factorial(n)*eval(M)
}

prob

[1] 0.0497938498 0.0003390956 0.0011304726
[4] 0.0028269513 0.0056569821 0.0094385700
[7] 0.0135130114 0.0169646392 0.0190127388
[10] 0.0193392713 0.0181755410
```

□

APPLICATION 6.2. An auto insurance company models the number of claims up to time t , $N(t)$, as a Poisson process with random intensity rate Λ which represents an accident-proneness index of a policyholder. The values of Λ are distributed according to a gamma distribution with mean α/β and variance α/β^2 .

(a) The expectation and variance of $N(t)$ are $\mathbb{E}(N(t)) = t\mathbb{E}(\Lambda) = \alpha t/\beta$, and $\text{Var}(N(t)) = t^2\text{Var}(\Lambda) + t\mathbb{E}(\Lambda) = \alpha t^2/\beta^2 + \alpha t/\beta$.

To compute these quantities for some specific values of the parameters, let's assume that $\alpha = 0.3$ and $\beta = 1$. The average total number of claims per policyholder that the company has to deal with every 5 years is $\mathbb{E}(N(5)) = (0.3)(5)/(1) = 1.5$. The standard deviation is

$$\sqrt{\text{Var}(N(5))} = \sqrt{(0.3)(5)^2/(1)^2 + (0.3)(5)/(1)} = 3.$$

(b) The marginal probability mass function of $N(t)$ has the form

$$\begin{aligned} \mathbb{P}(N(t) = n) &= \int_0^\infty \frac{(\lambda t)^n}{n!} e^{-\lambda t} \frac{\lambda^{\alpha-1} \beta^\alpha}{\Gamma(\alpha)} e^{-\beta \lambda} d\lambda \\ &= \frac{t^n}{n!} \frac{\Gamma(n+\alpha)}{\Gamma(\alpha)} \frac{\beta^\alpha}{(t+\beta)^{n+\alpha}} \int_0^\infty \frac{\lambda^{n+\alpha-1} (t+\beta)^{n+\alpha}}{\Gamma(n+\alpha)} e^{-(t+\beta)\lambda} d\lambda \\ &= \frac{(n+\alpha-1)!}{n! (\alpha-1)!} \left(\frac{\beta}{t+\beta} \right)^\alpha \left(1 - \frac{\beta}{t+\beta} \right)^n. \end{aligned}$$

Therefore, $N(t)$ has a *gamma-Poisson mixture distribution*, which has the algebraic form of a negative binomial distribution with parameters α and $p = \frac{\beta}{t + \beta}$. Since α assumes real values that are not restricted to integers, this is not a true negative binomial distribution. The mean and variance of this distribution are

$$\mathbb{E}(N(t)) = \frac{\alpha(1-p)}{p} = \frac{\alpha t/(t+\beta)}{\beta/(t+\beta)} = \alpha t/\beta,$$

and

$$\mathbb{V}ar(N(t)) = \frac{\alpha(1-p)}{p^2} = \frac{\alpha t/(t+\beta)}{\beta^2/(t+\beta)^2} = \frac{\alpha t(t+\beta)}{\beta^2} = \alpha t^2/\beta^2 + \alpha t/\beta.$$

These coincide with the expressions derived in part (a).

(c) We can also compute the conditional probability that Λ , the accident-proneness index of a policyholder, doesn't exceed some particular value λ , provided that the policyholder has made n claims within t years. We derive

$$\begin{aligned} \mathbb{P}(\Lambda \leq \lambda \mid N(t) = n) &= \frac{\mathbb{P}(N(t) = n, \Lambda \leq \lambda)}{\mathbb{P}(N(t) = n)} = \frac{\int_0^\lambda \frac{(ut)^n}{n!} e^{-ut} \frac{u^{\alpha-1}\beta^\alpha}{\Gamma(\alpha)} e^{-\beta u} du}{\int_0^\infty \frac{(ut)^n}{n!} e^{-ut} \frac{u^{\alpha-1}\beta^\alpha}{\Gamma(\alpha)} e^{-\beta u} du} \\ &= \frac{\int_0^\lambda u^{n+\alpha-1} e^{-(t+\beta)u} du}{\int_0^\infty u^{n+\alpha-1} e^{-(t+\beta)u} du} = \int_0^\lambda \frac{u^{n+\alpha-1} (t+\beta)^{n+\alpha}}{\Gamma(n+\alpha)} e^{-(t+\beta)u} du. \end{aligned}$$

This is a gamma distribution with parameters $n + \alpha$ and $t + \beta$. The expected value of Λ for a policyholder who is known to have made n claims within t years is $\mathbb{E}(\Lambda \mid N(t) = n) = \frac{n+\alpha}{t+\beta}$.

For $\alpha = 0.3$ and $\beta = 1$, the expected accident-proneness index for a policyholder who has made two claims in 5 years is $\mathbb{E}(\Lambda \mid N(5) = 2) = \frac{2+0.3}{5+1} = 0.3833$. \square

Exercises

EXERCISE 6.1. Let $\{N(t), t \geq 0\}$ be a conditional Poisson process with the random intensity rate Λ . Show that for any $t \geq s \geq 0$,

- (a) $\text{Cov}(N(s), N(t) - N(s)) = s(t-s)\mathbb{V}ar(\Lambda)$.
- (b) $\text{Cov}(N(s), N(t)) = st\mathbb{V}ar(\Lambda) + s\mathbb{E}(\Lambda)$.

EXERCISE 6.2. Suppose $\{N(t), t \geq 0\}$ is a conditional Poisson process and the random intensity rate Λ has density $f_\Lambda(\lambda)$, $\lambda > 0$. Show that

(a) The conditional cumulative distribution function of Λ , given $N(t) = n$, is

$$F_{\Lambda|N(t)}(\lambda|n) = \mathbb{P}(\Lambda \leq \lambda | N(t) = n) = \frac{\int_0^\lambda u^n e^{-ut} f_\Lambda(u) du}{\int_0^\infty u^n e^{-ut} f_\Lambda(u) du}.$$

(b) The conditional density function of Λ , given $N(t) = n$, is

$$f_{\Lambda|N(t)}(\lambda|n) = \frac{\lambda^n e^{-\lambda t} f_\Lambda(\lambda)}{\int_0^\infty \lambda^n e^{-\lambda t} f_\Lambda(\lambda) d\lambda}.$$

(c) The conditional expected value of Λ , given $N(t) = n$, is

$$\mathbb{E}[\Lambda | N(t) = n] = \frac{\int_0^\infty \lambda^{n+1} e^{-\lambda t} f_\Lambda(\lambda) d\lambda}{\int_0^\infty \lambda^n e^{-\lambda t} f_\Lambda(\lambda) d\lambda}.$$

EXERCISE 6.3. Suppose that 46% of all visitors of an amusement park are teenagers, 24% are adults, and the rest are kids. Assume that the number of visitors who come into the park during a busy hour can be modeled as a Poisson process with the random intensity rate that varies depending on age group: 4 per minute for teens, 2 per minute for adults, and 3 per minute for kids.

- Write down the model and specify all parameters. Find the mean and variance of the number of visitors within time t .
- Simulate five trajectories of the process with 200 visitors each.
- Simulate five trajectories of the process that depict arrivals within 1 hour.

EXERCISE 6.4. A credit union assigns to all its clients a rating value Λ in such a way that a client defaults on a credit account according to a Poisson process with rate Λ (per year). The distribution of Λ is uniform on $[0, 2]$.

- Find the average number of defaults that a client has within a 5-year period.
- Find the variance of the number of defaults within a 5-year period.
- Find the covariance between the number of defaults during the first 3 years and that during the subsequent 2 years. Hint: See Exercise 6.1(a).
- Find the covariance between the number of defaults during the first 3 years and that during the first 5 years. Hint: See Exercise 6.1(b).
- Find the probability that the client's rating value is less than 0.5, given that he has had two defaults within a 5-year period. Hint: See Exercise 6.2.

EXERCISE 6.5. Snow Water Equivalent (SWE) describes the amount of water contained within a snowpack, measured in inches. It is an important notion in environmental science, agriculture, and forestry. An annual SWE is well

modeled by a conditional Poisson process. The amount of SWE is a Poisson process with rate Λ inches per year that is itself a Poisson random variable with a rate of 24.3.

- (a) Compute the average and standard deviation of SWE for a 1-year period. For a 5-year period.
- (b) Simulate five trajectories that reach 140 inches of SWE each.
- (c) Simulate five trajectories spanning over a 7-year period.

EXERCISE 6.6. In the textile industry, a series issue is the number of defects in fabric per linear footage. Suppose quality control engineers model the number of defects as a conditional Poisson process with a random intensity rate Λ per yard for the fabric of standard width. The random variable Λ has a gamma distribution with a mean of 0.07 and a standard deviation of 0.01.

- (a) Compute the expected number of defects in a 40-yard roll of fabric. Find the standard deviation of the number of defects in the roll.
- (b) Given that four defects were found in a 40-yard roll of fabric, find the probability that Λ exceeds 0.08.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Birth-and-Death Process

7.1 Definition of Birth-and-Death Process

A continuous-time Markov chain $\{X(t), t \geq 0\}$ with state space $S = \{0, 1, 2, \dots\}$ is called a *birth-and-death process*¹ if, given that the chain is in state n , the time to transition to state $n + 1$ is exponentially distributed with mean $1/\lambda_n$, and the time to transition to state $n - 1$ is exponentially distributed with mean $1/\mu_n$. The two waiting times are independent.

PROPOSITION 7.1. In a birth-and-death process, the transition probabilities are $P_{0,1} = 1$, $P_{n,n+1} = \frac{\lambda_n}{\lambda_n + \mu_n}$, and $P_{n,n-1} = \frac{\mu_n}{\lambda_n + \mu_n}$. All the other transition probabilities are 0's. The one-step transition probability matrix has the form

$$\mathbf{P} = \begin{array}{c} \begin{array}{ccccc} & 0 & 1 & 2 & 3 & \dots \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ \dots \end{array} & \begin{bmatrix} 0 & 1 & 0 & 0 & \dots \\ \frac{\mu_1}{\lambda_1 + \mu_1} & 0 & \frac{\lambda_1}{\lambda_1 + \mu_1} & 0 & \dots \\ 0 & \frac{\mu_2}{\lambda_2 + \mu_2} & 0 & \frac{\lambda_2}{\lambda_2 + \mu_2} & \dots \\ 0 & 0 & \frac{\mu_3}{\lambda_3 + \mu_3} & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix} \end{array} \end{array}.$$

PROOF: Consider two independent exponential random variables T_B and T_D with means $1/\lambda$ and $1/\mu$, respectively. The variable T_B represents the waiting time until a “birth,” and T_D represents the waiting time until a “death.” A “birth” occurs before a “death,” if $T_B < T_D$. The probability of this event is

$$\mathbb{P}(T_B < T_D) = \int_0^\infty \int_x^\infty \lambda e^{-\lambda x} \mu e^{-\mu y} dy dx = \int_0^\infty \lambda e^{-\lambda x} e^{-\mu x} dx = \frac{\lambda}{\lambda + \mu}.$$

¹The first example of a birth-and-death process was described in 1939 by William Feller, a renown Croatian-American mathematician, in “Die Grundlagen der Volterraschen Theorie des Kampfes ums Dasein in wahrscheinlichkeitstheoretischer Behandlung,” *Acta Biotheoretica*, 5: 11 – 40.

Analogously, a “death” occurs before a “birth” if $T_D < T_B$, which happens with the complementary probability

$$\mathbb{P}(T_D < T_B) = 1 - \mathbb{P}(T_B < T_D) = 1 - \frac{\lambda}{\lambda + \mu} = \frac{\mu}{\lambda + \mu}. \quad \square$$

PROPOSITION 7.2. Suppose a birth-and-death process is in state n . Then the waiting time until a transition occurs is exponential with mean $1/(\lambda_n + \mu_n)$.

PROOF: Referring to the proof of the previous proposition, we see that a transition occurs at time T_B or time T_D , whichever happens first. That is, we need to find the distribution of $\min(T_B, T_D)$. We write

$$\begin{aligned} F_{\min(T_B, T_D)}(t) &= \mathbb{P}(T_B \leq t, T_B < T_D) + \mathbb{P}(T_D \leq t, T_D < T_B) \\ &= \int_0^t \int_x^\infty \lambda e^{-\lambda x} \mu e^{-\mu y} dy dx + \int_0^t \int_y^\infty \lambda e^{-\lambda x} \mu e^{-\mu y} dx dy \\ &= \int_0^t \lambda e^{-\lambda x} e^{-\mu x} dx + \int_0^t e^{-\lambda y} \mu e^{-\mu y} dy \\ &= \int_0^t (\lambda + \mu) e^{-(\lambda + \mu)u} du = 1 - e^{-(\lambda + \mu)t}, \end{aligned}$$

which is an exponential distribution with mean $1/(\lambda + \mu)$. \square

PROPOSITION 7.3. Consider a birth-and-death process, and denote by $P_n(t)$ the probability that at time t the process is in state n . The probabilities $P_n(t)$, $n = 0, 1, \dots$, satisfy the system of the *Kolmogorov forward equations*:

$$\begin{cases} P_0'(t) = -\lambda_0 P_0(t) + \mu_1 P_1(t), \\ P_n'(t) = \lambda_{n-1} P_{n-1}(t) + \mu_{n+1} P_{n+1}(t) - (\lambda_n + \mu_n) P_n(t), \quad n = 1, 2, \dots, \end{cases} \quad (7.1)$$

with the boundary condition $P_{n_0}(0) = 1$.

PROOF: We will omit the rigorous derivation of these equations but will explain their simple meaning. How can the process transition into state n ? Only if there are $n-1$ particles and one more is born, or there are $n+1$ particles and one dies. That is, $\lambda_{n-1} P_{n-1}(t) + \mu_{n+1} P_{n+1}(t)$ represents the rate of change with respect to time of probability to transition into state n . By the same token, $(\lambda_n + \mu_n) P_n(t)$ gives the rate of change over time of probability to transition out of state n (one particle is born or dies). Thus, the expression on the right is the difference between the in and out rates, and so is the meaning of the derivative $P_n'(t)$ on the left. \square

REMARK 7.1. The mean and variance of $\{X(t), t \geq 0\}$, a birth-and-death process, are computed as

$$\mathbb{E}(X(t)) = \sum_{n=0}^{\infty} n P_n(t), \text{ and } \mathbb{V}ar(X(t)) = \sum_{n=0}^{\infty} n^2 P_n(t) - [\mathbb{E}(X(t))]^2. \quad \square$$

EXAMPLE 7.1. A Poisson process is an example of a birth-and-death process with $\lambda_n = \lambda$ and $\mu_n = 0$, for $n = 0, 1, \dots$. Because there are no “deaths,” it is a *pure birth process*.

(a) We know that in this process, times until “births” are independent and exponentially distributed with mean $1/\lambda_n = 1/\lambda$.

(b) The transition probabilities are $P_{0,1} = 1$, $P_{n,n+1} = \lambda_n/(\lambda_n + \mu_n) = \lambda/(\lambda + 0) = 1$, and $P_{n,n-1} = \mu_n/(\lambda_n + \mu_n) = 0/(\lambda + 0) = 0$, which we know is true since only jumps of size +1 are admissible in a Poisson process.

(c) The probabilities $P_n(t)$, $n = 0, 1, \dots$, satisfy the Kolmogorov forward equations $P'_0(t) = -\lambda P_0(t)$ and $P'_n(t) = \lambda P_{n-1}(t) - \lambda P_n(t)$, for $n = 1, 2, \dots$, with the boundary condition $P_0(0) = 1$. The solution to these equations is the Poisson probability mass function $P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$, $n = 0, 1, \dots$, which can be verified by writing

$$P'_0(t) = (e^{-\lambda t})' = -\lambda e^{-\lambda t} = -\lambda P_0(t),$$

and for $n \geq 1$,

$$\begin{aligned} P'_n(t) &= \left(\frac{(\lambda t)^n}{n!} e^{-\lambda t} \right)' = \frac{\lambda^n}{n!} (n t^{n-1} - \lambda t^n) e^{-\lambda t} = \lambda \frac{(\lambda t)^{n-1}}{(n-1)!} e^{-\lambda t} \\ &\quad - \lambda \frac{(\lambda t)^n}{n!} e^{-\lambda t} = \lambda P_{n-1}(t) - \lambda P_n(t), \text{ and } P_0(0) = \frac{0^0}{0!} e^0 = 1. \quad \square \end{aligned}$$

(d) Both the mean and variance of the Poisson process, as we know, are equal to λt .

EXAMPLE 7.2. A *linear birth-and-death process* $\{X(t), t \geq 0\}$ is a birth-and-death process with the parameters $\lambda_n = n\lambda$ and $\mu_n = \mu n$, $n = 0, 1, \dots$. Note that this process state 0 is an absorbing state. We assume that the process starts in state 1.

(a) In this process, times until “births” and “deaths” are independent and exponentially distributed with mean $1/\lambda_n = 1/(n\lambda)$ and $1/\mu_n = 1/(n\mu)$, respectively. Time until a transition (“birth” or “death”) are independent exponentially distributed random variables with mean $1/(\lambda_n + \mu_n) = 1/[n(\lambda + \mu)]$.

(b) The transition probabilities are $P_{0,1} = 0$, $P_{n,n+1} = \lambda_n/(\lambda_n + \mu_n) = \lambda/(\lambda + \mu)$, and $P_{n,n-1} = \mu_n/(\lambda_n + \mu_n) = \mu/(\lambda + \mu)$.

(c) The Kolmogorov forward equations in this case assume the form: $P_0'(t) = \mu P_1(t)$ and for $n = 1, 2, \dots$, $P_n'(t) = (n-1)\lambda P_{n-1}(t) + (n+1)\mu P_{n+1}(t) - n(\lambda + \mu)P_n(t)$, with the initial condition $P_1(0) = 1$. The solution of these equations can be written as

$$P_0(t) = P_0 = \frac{\mu e^{(\lambda-\mu)t} - \mu}{\lambda e^{(\lambda-\mu)t} - \mu},$$

and

$$P_n(t) = (1 - P_0)\left(1 - \frac{\lambda}{\mu} P_0\right)\left(\frac{\lambda}{\mu} P_0\right)^{n-1}, \quad n = 1, 2, \dots$$

This distribution is a mixture of a point mass at zero and a geometric distribution modeling the number of trials until the first success where the probability of success is $p = 1 - \frac{\lambda}{\mu} P_0$.

(d) The mean of the process can be computed as

$$\mathbb{E}(X(t)) = (1 - P_0) \frac{1}{p} = \frac{1 - P_0}{1 - \frac{\lambda}{\mu} P_0} = e^{(\lambda-\mu)t}.$$

The variance is equal to

$$\mathbb{V}ar(X(t)) = (1 - P_0) \frac{1-p}{p^2} = \frac{\lambda + \mu}{\lambda - \mu} e^{(\lambda-\mu)t} (e^{(\lambda-\mu)t} - 1). \quad \square$$

7.2 Simulations in R

SIMULATION 7.1. Below we simulate a 20-step trajectory of a linear birth-and-death process with parameters $\lambda = 0.3$ and $\mu = 0.1$ that starts at time 0 in state 1. We generate two independent exponential times with rates $1/(n\lambda)$

and $1/(n\mu)$, and transition 1 unit up if a “birth” occurs before “death” or 1 unit down, otherwise.

```
#specifying parameters
lambda<- 0.3
mu<- 0.1
njumps<- 20

#setting state and time as vectors
N<- c()
time<- c()

#setting initial values
N[1]<- 1
time[1]<- 0

#specifying seed
set.seed(1022171)

#simulating trajectory
i<- 2

repeat {
time.birth<- (-1/(N[i-1]*lambda))*log(runif(1))
  time.death<- (-1/(N[i-1]*mu))*log(runif(1))

if(time.birth < time.death | N[i-1]==0) {
  time[i]<- time[i-1]+time.birth-0.001
  N[i]<- N[i-1]

  if(i==2*njumps+2) break
  else {
    time[i+1]<- time[i]+0.001
    N[i+1]<- N[i]+1
    i<- i+2
  }
}
```



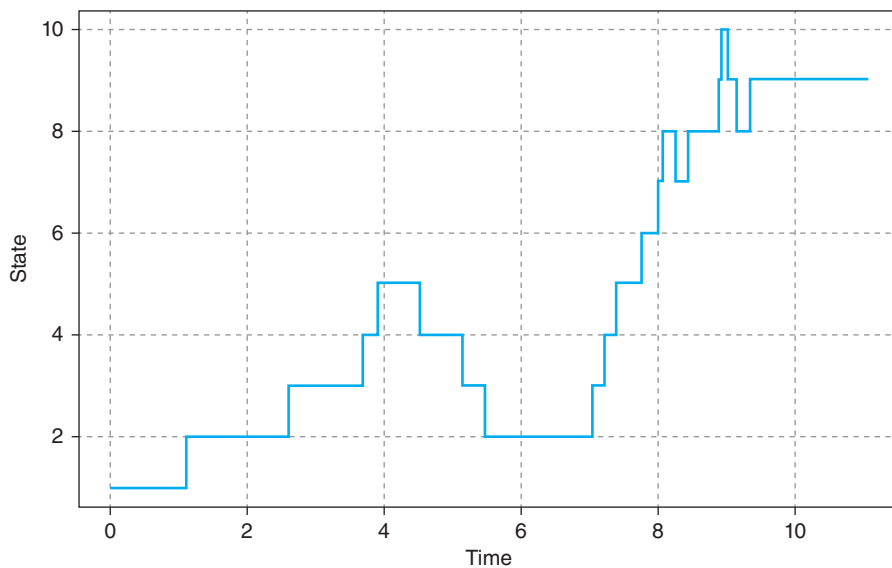
```

if(time.death < time.birth & N[i-1]!=0) {
  time[i]<- time[i-1]+time.death-0.001
  N[i]<- N[i-1]
  if(i==2*njumps+2) break
  else {
    time[i+1]<- time[i]+0.001
    N[i+1]<- N[i]-1
    i<- i+2
  }
}

}

#plotting trajectory
plot(time, N, type="l", lty=1, lwd=2, col=4, xlab="Time",
      ylab="State", panel.first=grid())

```



□

7.3 Applications of Birth-and-Death Process

APPLICATION 7.1. An $M/M/1$ queue is a birth-and-death process $\{X(t), t \geq 0\}$ with $\lambda_n = \lambda$ and $\mu_n = \mu$. We assume $\lambda < \mu$. In this process, customers join a queue (representing “births”) at independent exponential times with mean $1/\lambda$, and leave the system (representing “deaths”) after going through the service, which time is exponentially distributed with mean $1/\mu$. All customers act independently. In the name of the process, the first “M” means that the customer arrival process is Markovian, the second “M” refers to the fact that the service time is Markovian, and the “1” stands for a single server. An example of $M/M/1$ process is the number of customers in a bank with a single bank teller: customers enter the bank and join the line to the bank teller, then when it is their turn, they get a service from the teller and leave the bank. Another example is the number of broken cars in a repair shop with a single repairman: broken cars are added to the line to get a service from the repairman, and leave the shop once they are repaired.

(a) The Kolmogorov forward equations for $M/M/1$ process are

$$\begin{cases} P_0'(t) = -\lambda P_0(t) + \mu P_1(t), \\ P_n'(t) = \lambda P_{n-1}(t) + \mu P_{n+1}(t) - (\lambda + \mu) P_n(t), \quad n = 1, 2, \dots, \end{cases}$$

with the boundary condition $P_0(0) = 1$. The solution to these equations exists but is rarely used in practice. Instead, the *limiting* (or *steady-state*) probabilities are computed. They are defined as $\lim_{t \rightarrow \infty} P_n(t) = P_n$, $n \geq 0$. To find the limiting probabilities, we pass to the limit in the Kolmogorov forward equations as t tends to infinity. Replacing the left-hand side by 0 (since the derivative of a constant is 0), we obtain

$$\begin{cases} 0 = -\lambda P_0 + \mu P_1, \\ 0 = \lambda P_{n-1} + \mu P_{n+1} - (\lambda + \mu) P_n, \quad n = 1, 2, \dots \end{cases}$$

These can be rewritten in the form of what is known as the *balance* (or *equilibrium*) equations:

$$\begin{cases} \mu P_1 = \lambda P_0, \\ (\lambda + \mu) P_n = \lambda P_{n-1} + \mu P_{n+1}, \quad n = 1, 2, \dots \end{cases}$$

The expression on the left represents the mean rate of leaving the state n , whereas the right-hand side gives the mean rate of entering state n . Thus, the balance equations equate the mean departure rate and the mean entrance rate.

To solve the balance equations, we notice that $\mu P_1 - \lambda P_0 = 0$ and $\mu P_{n+1} - \lambda P_n = \mu P_n - \lambda P_{n-1} = \dots = \mu P_1 - \lambda P_0 = 0$. Therefore,

$P_{n+1} = \frac{\lambda}{\mu} P_n = \left(\frac{\lambda}{\mu}\right)^2 P_{n-1} = \dots = \left(\frac{\lambda}{\mu}\right)^{n+1} P_0, \quad n = 0, 1, \dots$ Since all the probabilities must add up to 1, we conclude that $P_0 = \left[\sum_{n=0}^{\infty} \left(\frac{\lambda}{\mu}\right)^n \right]^{-1} = 1 - \frac{\lambda}{\mu}$, and so $P_n = \left(1 - \frac{\lambda}{\mu}\right) \left(\frac{\lambda}{\mu}\right)^n, \quad n = 0, 1, 2, \dots$ This is a geometric distribution modeling the number of failures before the first success, where the success probability is $p = 1 - \lambda/\mu$.

(b) The average number of customers in the system, in the long run, is computed as

$$\lim_{t \rightarrow \infty} \mathbb{E}(X(t)) = \frac{1-p}{p} = \frac{1 - (1 - \lambda/\mu)}{1 - \lambda/\mu} = \frac{\lambda}{\mu - \lambda}.$$

(c) We will show that the amount of time T a customer spends in the system is an exponential random variable with mean $1/(\mu - \lambda)$. Indeed, suppose when the customer arrives, there are already n customers in the system. If $n = 0$, then T is the service time which is exponential with mean $1/\mu$. If $n > 0$, the customer will have to wait for one customer to complete the service, and then for n additional customers (including himself) to go through the service. Using the memoryless property of an exponential distribution, we conclude that the waiting time, in this case, is the sum of $n + 1$ independent exponential with mean $1/\mu$ random variables, which is a random variable having a gamma distribution with mean $(n + 1)/\mu$. We write

$$\begin{aligned} f_T(t) &= \mathbb{E}[f_T(t) | n \text{ customers}] = \sum_{n=0}^{\infty} \frac{t^n \mu^{n+1}}{\Gamma(n+1)} e^{-\mu t} P_n \\ &= \sum_{n=0}^{\infty} \frac{(\mu t)^n}{n!} \mu e^{-\mu t} \left(1 - \frac{\lambda}{\mu}\right) \left(\frac{\lambda}{\mu}\right)^n = (\mu - \lambda) e^{-\mu t} \sum_{n=0}^{\infty} \frac{(\lambda t)^n}{n!} = (\mu - \lambda) e^{-(\mu - \lambda)t}, \end{aligned}$$

which is an exponential distribution with mean $1/(\mu - \lambda)$. From here, we can conclude that the probability that a customer spends more than time t in the system is $\mathbb{P}(T > t) = e^{-(\mu - \lambda)t}$.

(d) Consider an M/M/1 system in which arrivals occur with the rate $\lambda = 1$ per minute, and departures occur with the rate $\mu = 1.5$ per minute. The steady-state probabilities are

$$P_n = \left(1 - \frac{1}{1.5}\right) \left(\frac{1}{1.5}\right)^n = \left(\frac{1}{3}\right) \left(\frac{2}{3}\right)^n, \quad n = 0, 1, \dots$$

In the long run, there will be, on average, $\frac{\lambda}{\mu - \lambda} = \frac{1}{1.5 - 1} = 2$ customers in the system. The probability that in a long run, a customer will spend over 5 minutes in the system is $P(T > 5) = e^{-(1.5-1)(5)} = 0.082085$. \square

Exercises

EXERCISE 7.1. A Yule process² (or a linear birth process) $\{X(t), t \geq 0\}$ is a birth-and-death process with $\lambda_n = n\lambda$ and $\mu_n = 0$, for all $n \geq 0$. In this process, each particle gives birth to one particle, independently of others, and never dies. It is a pure birth process.

(a) Suppose that initially the process is in state 1 (i.e., there is a single particle in the system). Show that the Kolmogorov forward equations (7.1) have the form $P_1'(t) = -\lambda P_1(t)$ and $P_n'(t) = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t)$, $n = 2, 3, \dots$, with the boundary condition $P_1(0) = 1$.

(b) Verify that $P_n(t) = e^{-\lambda t} (1 - e^{-\lambda t})^{n-1}$, $n = 1, 2, \dots$. Note that it is a geometric distribution that models the number of trials until the first success where the probability of a success is $p = e^{-\lambda t}$.

(c) Show that the mean of the Yule process at time t is $\mathbb{E}(X(t)) = e^{\lambda t}$, and the variance is $\text{Var}(X(t)) = e^{\lambda t}(e^{\lambda t} - 1)$.

(d) If $\lambda = 4$ per week, what is the probability that there will be between 3 and 5 particles at week 1? What is the mean and standard deviation of the number of particles at week 1?

EXERCISE 7.2. Consider a Yule process, a birth-and-death process with parameters $\lambda_n = n\lambda$ and $\mu_n = 0$, for all $n \geq 0$. Suppose initially the process is in state m . That is, the initial size of the population is m particles.

(a) Verify that the Kolmogorov forward equations (7.1) have the form $P_m'(t) = -m\lambda P_m(t)$ and $P_n'(t) = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t)$, $n = m, m+1, \dots$, with the boundary condition $P_m(0) = 1$.

(b) Verify that $P_n(t) = \binom{n-1}{n-m} e^{-m\lambda t} (1 - e^{-\lambda t})^{n-m}$, $n = m, m+1, \dots$.

Note that it is a negative binomial distribution of the number of trials until the m th success, where the probability of a success is $p = e^{-\lambda t}$.

(c) Show that the mean of the Yule process at time t is $\mathbb{E}(X(t)) = m e^{\lambda t}$, and the variance is $\text{Var}(X(t)) = m e^{\lambda t}(e^{\lambda t} - 1)$.

(d) If there are originally 5 particles in the population and they multiply with rate $\lambda = 0.2$ per day, what is the probability that there will be exactly 12 particles on day 2? What are the average and standard deviation of the number of particles on day 2?

²Proposed in Yule, G. U. (1925). "A mathematical theory of evolution, based on the conclusions of Dr. J. C. Willis, F.R.S." *Philosophical transactions of the Royal Society of London. Series B, containing papers of a biological character*, 213: 21–87.

EXERCISE 7.3. A *linear death process* $\{X(t), t \geq 0\}$ is a birth-and-death process with the initial state N , and parameters $\lambda_n = 0$ and $\mu_n = n\mu$ for $n = 0, 1, \dots, N-1$.

(a) Show that the probabilities $P_n(t)$, $n = 0, 1, \dots, N$, satisfy the Kolmogorov forward equations

$$\begin{cases} P'_N(t) = -N\mu P_N(t), \\ P'_n(t) = (n+1)\mu P_{n+1}(t) - n\mu P_n(t), \quad n = 0, 1, \dots, N-1. \end{cases}$$

(b) Verify that $P_n(t) = \binom{N}{n} (e^{-\mu t})^n (1 - e^{-\mu t})^{N-n}$, $n = 0, 1, \dots, N$, which is a binomial distribution with parameters N and $p = e^{-\mu t}$.

(c) Show that the mean and variance of this process at time t are $\mathbb{E}(X(t)) = N e^{-\mu t}$ and $\text{Var}(X(t)) = N e^{-\mu t} (1 - e^{-\mu t})$.

(d) Assume $\mu = 0.02$ and $N = 15$. Find the probability that the process is in state 12 at time 3. What is the expected state at time 3? What is its standard deviation?

EXERCISE 7.4. Consider a linear birth-and-death process $\{X(t), t \geq 0\}$ described in Example 7.2.

(a) Suppose $\lambda = 1.3$ and $\mu = 0.2$. Compute the probability that the process will be in state 4 at time 2. Find the mean and variance of the process at time 2.

(b) Simulate a 50-step trajectory of this process, assuming $\lambda = 1.3$, $\mu = 0.2$, and the initial state is 1.

EXERCISE 7.5. Consider an M/M/1 queue described in Application 7.1.

(a) In the long run, how many customers do we expect to see in the system if $\lambda > \mu$? Explain on an intuitive level.

(b) Assume $\lambda = 3$ and $\mu = 5$. Find the probability that there will be more than 2 customers in the system in the long run.

(c) Compute the average number of customers in the system in the long run. Use $\lambda = 3$ and $\mu = 5$.

(d) Find the proportion of customers in the system in the long run who have to wait more than 1 minute.

EXERCISE 7.6. Suppose the bird count in a flock can be modeled as a birth-and-death process with *immigration* and *emigration*. In this model, $\lambda_n = n\lambda + \alpha$ and $\mu_n = n\mu + \beta$ where α is the rate of immigration (joining the flock), and β is the rate of emigration (leaving the flock). Generate a trajectory of the process until the flock size increases from 10 to 25 birds, assuming $\lambda = 1$, $\alpha = 0.3$, $\beta = 0.1$ and

- (a) $\mu = 0.2$.
- (b) $\mu = 0.8$.
- (c) $\mu = 1$.
- (d) $\mu = 1.2$.
- (e) Discuss the difference in behaviors of the four trajectories. How many time units does each one span? Does the flock ever die out?



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Branching Process

8.1 Definition of Branching Process

A discrete-time stochastic process $\{X_n, n \geq 0\}$ that gives the size of the n th generation of multiplying particles is called a *branching process* (or the *Bienaymé-Galton-Watson process*¹). It starts with X_0 particles in the 0th generation. Each particle survives for one-time unit, at the end of which it splits into a random number of particles with a known probability distribution. The offspring particles survive for one-time unit, and produce a random number of offspring, independently from each other, and the process continues. Put

formally, $X_n = \sum_{i=1}^{X_{n-1}} Z_i$ where Z_i is the size of the offspring of the i th particle in the $(n-1)$ st generation. The distribution of the offspring size Z_i is identical for all particles, with $p_k = \mathbb{P}(Z_i = k)$, $k = 0, 1, \dots$, $\mathbb{E}(Z_i) = \mu$, and $\text{Var}(Z_i) = \sigma^2$.

PROPOSITION 8.1. A branching process is a Markov chain.

PROOF: To prove that the Markov property holds, we use the fact that the distribution of the number of offspring for each individual particle is independent of the size of the current and all previous generations. We write

$$\begin{aligned} & \mathbb{P}(X_n = j_n \mid X_0 = j_0, X_1 = j_1, \dots, X_{n-1} = j_{n-1}) \\ &= \mathbb{P}\left(\sum_{i=1}^{j_{n-1}} Z_i = j_n \mid X_0 = j_0, X_1 = j_1, \dots, X_{n-1} = j_{n-1}\right) \end{aligned}$$

¹A French statistician Jules Bienaymé first addressed the problem of survival of family names in his 1845 article “De la loi de multiplication et de la durée des familles.” *Société Philomatique de Paris. Extraits des Procès-Verbaux des Séances*: 37 – 39. This problem was rediscovered later by Englishman Sir Francis Galton who posed the problem (under the number 4001) in *The Educational Times and Journal of the College of Receptors*, Vol. XXV, No. 143, on page 300. The solution by English mathematician Reverend Henry William Watson was published in the same journal, Vol. XXVI, No. 148, on page 115.

$$= \mathbb{P}\left(\sum_{i=1}^{j_{n-1}} Z_i = j_n \mid X_{n-1} = j_{n-1}\right) = \mathbb{P}(X_n = j_n \mid X_{n-1} = j_{n-1}). \quad \square$$

PROPOSITION 8.2. Consider a branching process with a single initial ancestor, $X_0 = 1$.

- (a) The mean of the size of the n th generation is $\mathbb{E}(X_n) = \mu^n$.
 (b) The variance is $\mathbb{V}ar(X_0) = 0$, $\mathbb{V}ar(X_1) = \sigma^2$, and for $n \geq 2$,

$$\mathbb{V}ar(X_n) = \begin{cases} \sigma^2 \mu^{n-1} \left(\frac{1-\mu^n}{1-\mu} \right), & \text{if } \mu \neq 1 \\ \sigma^2 n, & \text{if } \mu = 1. \end{cases}$$

PROOF: (a) To find the expected value of X_n , we condition on the value of the size of the previous generation X_{n-1} . We write

$$\begin{aligned} \mathbb{E}(X_n) &= \mathbb{E}[\mathbb{E}(X_n \mid X_{n-1})] = \mathbb{E}\left[\mathbb{E}\left(\sum_{i=1}^{X_{n-1}} Z_i \mid X_{n-1}\right)\right] \\ &= \mathbb{E}(X_{n-1} \mathbb{E}(Z_i)) = \mu \mathbb{E}(X_{n-1}) = \mu^2 \mathbb{E}(X_{n-2}) = \cdots = \mu^n \mathbb{E}(X_0) = \mu^n. \end{aligned}$$

(b) The variance of $X_0 = 1$ is 0. The variance of $X_1 = Z_1$ is σ^2 . To compute the expression for the variance of X_n for $n \geq 2$, we condition on the value of X_{n-1} . We obtain

$$\begin{aligned} \mathbb{V}ar(X_n) &= \mathbb{V}ar[\mathbb{E}(X_n \mid X_{n-1})] + \mathbb{E}[\mathbb{V}ar(X_n \mid X_{n-1})] \\ &= \mathbb{V}ar\left[\mathbb{E}\left(\sum_{i=1}^{X_{n-1}} Z_i \mid X_{n-1}\right)\right] \\ &+ \mathbb{E}\left[\mathbb{V}ar\left(\sum_{i=1}^{X_{n-1}} Z_i \mid X_{n-1}\right)\right] = \mathbb{V}ar[X_{n-1} \mathbb{E}(Z_i)] + \mathbb{E}[X_{n-1} \mathbb{V}ar(Z_i)] \\ &= \mathbb{V}ar(\mu X_{n-1}) + \mathbb{E}(\sigma^2 X_{n-1}) = \mu^2 \mathbb{V}ar(X_{n-1}) + \sigma^2 \mu^{n-1} \\ &= \mu^2 [\mu^2 \mathbb{V}ar(X_{n-2}) + \sigma^2 \mu^{n-2}] + \sigma^2 \mu^{n-1} = \mu^4 \mathbb{V}ar(X_{n-2}) + \sigma^2 (\mu^{n-1} + \mu^n) \\ &= \cdots = \mu^{2n-2} \mathbb{V}ar(X_1) + \sigma^2 (\mu^{n-1} + \cdots + \mu^{2n-3}) = \sigma^2 (\mu^{n-1} + \cdots + \mu^{2n-2}) \\ &= \sigma^2 \mu^{n-1} (1 + \mu + \cdots + \mu^{n-1}) = \sigma^2 \mu^{n-1} \left(\frac{1-\mu^n}{1-\mu} \right), \quad \text{if } \mu \neq 1, \end{aligned}$$

and if $\mu = 1$, the variance is equal to $\sigma^2 n$. \square

EXAMPLE 8.1. (a) Suppose that the distribution of the size of the offspring is *Bernoulli*($\frac{1}{2}$). We know that $\mu = \frac{1}{2}$ and $\sigma^2 = \frac{1}{4}$. Therefore, the expected

size of the n th generation is $\mathbb{E}(X_n) = \left(\frac{1}{2}\right)^n$ with the standard deviation $\sqrt{\mathbb{V}ar(X_n)} = \sqrt{\left(\frac{1}{4}\right)\left(\frac{1}{2}\right)^{n-1} \left(\frac{1 - \left(\frac{1}{2}\right)^n}{1 - \frac{1}{2}}\right)} = \sqrt{\left(\frac{1}{2}\right)^n \left(1 - \left(\frac{1}{2}\right)^n\right)}.$

(b) Assume now that the distribution of the offspring size is *Binomial*(2, 1/2). It means that $\mu = 1$ and $\sigma^2 = 1/2$. Hence, $\mathbb{E}(X_n) = 1$, and the standard deviation is $\sqrt{\mathbb{V}ar(X_n)} = \sqrt{\frac{n}{2}}.$

(c) Here we will assume that the distribution of Z_i 's is *Binomial*(3, 1/2). Thus, the mean $\mu = 3/2$ and variance is $\sigma^2 = 3/4$. The mean of the size of the n th generation, therefore, is found as $\mathbb{E}(X_n) = (3/2)^n$ and the standard deviation is $\sqrt{\mathbb{V}ar(X_n)} = \sqrt{\left(\frac{3}{4}\right)\left(\frac{3}{2}\right)^{n-1} \left(\frac{1 - \left(\frac{3}{2}\right)^n}{1 - \frac{3}{2}}\right)} = \sqrt{\left(\frac{3}{2}\right)^n \left(\left(\frac{3}{2}\right)^n - 1\right)}. \quad \square.$

Branching processes are classified according to the value of the mean size of the offspring. If $\mu < 1$, the process is called *subcritical*, if $\mu = 1$, it is called *critical*, and if $\mu > 1$, it is called *supercritical*.

The *probability of extinction* is defined as the probability that a branching process with one initial ancestor will eventually have no particles. Put mathematically, let π_0 denote the probability of extinction. Then

$$\pi_0 = \lim_{n \rightarrow \infty} \mathbb{P}(X_n = 0 \mid X_0 = 1).$$

PROPOSITION 8.3. (a) For a subcritical ($\mu < 1$) process, the probability of extinction is 1 ($\pi_0 = 1$). Intuitively, if not enough particles are being born, the population will surely go extinct.

(b) For a critical ($\mu = 1$) process, $\pi_0 = 1$ (so the population is guaranteed to become extinct), unless $Z_i = 1$. In this case, the population consists of a single particle throughout all generations.

(c) For a supercritical ($\mu > 1$) process, the extinction can happen with a positive probability, but this probability is less than 1. In fact, π_0 is the smallest positive solution of the equation

$$\pi_0 = \sum_{k=0}^{\infty} \mathbb{P}(\text{extinction} \mid X_1 = k) p_k = \sum_{k=0}^{\infty} \pi_0^k p_k.$$

Note that $\pi_0 = 1$ is always one of the roots, which helps reduce this equation by one degree in case it has a polynomial form. \square

EXAMPLE 8.2. Going back to Example 8.1, we see that when the distribution of the offspring size is *Bernoulli*(1/2) ($\mu = 1/2$) or *Binomial*(2, 1/2) ($\mu = 1$), the population will go extinct with probability 1. However, in the case of *Binomial*(3, 1/2), the extinction of the population is not a sure thing. It will happen with probability π_0 where π_0 is the smallest positive root of $\pi_0 = \pi_0^0 (1/2)^3 + \pi_0^1 (3)(1/2)^3 + \pi_0^2 (3)(1/2)^3 + \pi_0^3 (1/2)^3$, or, rewritten in the standard form, $\pi_0^3 + 3\pi_0^2 - 5\pi_0 + 1 = 0$. Since $\pi_0 = 1$ is a solution, the cubic equation is reduced to the quadratic one $\pi_0^2 + 4\pi_0 - 1 = 0$, which roots are $-2 - \sqrt{5}$ and $\sqrt{5} - 2$. The probability of extinction is the positive of the two roots, $\pi = \sqrt{5} - 2 = 0.236068$. \square

EXAMPLE 8.3. Suppose a branching process $\{X_n, n \geq 0\}$ starts with a single particle and the particles multiply according to a geometric distribution with the probability mass function $p(x) = p(1-p)^x$, $x = 0, 1, 2, \dots$

(a) The mean of the geometric distribution is $\mu = \mathbb{E}(Z_n) = \frac{1-p}{p}$. The process is subcritical if $\mu < 1$, or $\frac{1-p}{p} < 1$, or $p > 0.5$. The process is critical if $\mu = 1$ or $p = 0.5$. The process is supercritical if $\mu > 1$, or $p < 0.5$.

(b) The average size of the n th generation is $\mathbb{E}(X_n) = \mu^n = \left(\frac{1-p}{p}\right)^n$. The variance of the geometric distribution is $\sigma^2 = \frac{1-p}{p^2}$, and hence, if $p \neq 0.5$,

$$\begin{aligned} \mathbb{V}ar(X_n) &= \sigma^2 \mu^{n-1} \left(\frac{1-\mu^n}{1-\mu} \right) = \frac{1-p}{p^2} \left(\frac{1-p}{p} \right)^{n-1} \left(\frac{1 - \left(\frac{1-p}{p} \right)^n}{1 - \frac{1-p}{p}} \right) \\ &= \frac{1}{2p-1} \left(\frac{1-p}{p} \right)^n \left(1 - \left(\frac{1-p}{p} \right)^n \right). \end{aligned}$$

If $p = 0.5$, $\mathbb{V}ar(X_n) = \sigma^2 n = \frac{(1-p)n}{p^2} = 2n$.

(c) To find the probability of extinction as a function of p , $0 < p < 0.5$, we note that π_0 is the smallest positive solution of the equation

$$\pi_0 = \sum_{n=0}^{\infty} \pi_0^n p (1-p)^n = p \sum_{n=0}^{\infty} (\pi_0(1-p))^n = \frac{p}{1 - \pi_0(1-p)}.$$

This is a quadratic equation $(\pi_0 - 1)(\pi_0 - \frac{p}{1-p}) = 0$. Therefore, $\pi_0 = \frac{p}{1-p}$. For $p \geq 0.5$, $\pi_0 = 1$. \square

8.2 Simulations in R

SIMULATION 8.1. Here we simulate generation sizes for subcritical, critical, and supercritical processes. We assume that each population starts with a single initial particle and the respective offspring distributions are binomial with parameters $n = 3$, $p = 0.2$ ($\mu = np = 0.6$), $n = 5$, $p = 0.2$ ($\mu = 1$), and $n = 3$, $p = 0.6$ ($\mu = 1.8$).

```
#subcritical branching process, mu=0.6
k<- 1
N<- c()
N[1]<- 1

set.seed(300168)
for (i in 2:100) {
  N[i]<- sum(rbinom(N[i-1],3,0.2))
  if (N[i]==0) {
    break }
}

N
```

```
[1] 1 2 2 1 0
```

```
#critical branching process, mu=1
N<- c()
N[1]<- 1

set.seed(3554218)
for (i in 2:100) {
  N[i]<- sum(rbinom(N[i-1],5,0.2))
  if (N[i]==0) {
    break }
}

N
```

```
[1] 1 3 6 6 10 6 7 2 3 3 2 1
[13] 1 0
```

```
#supercritical branching process, mu=1.8
N<- c()
N[1]<- 1
```

```
set.seed(965823)
for (i in 2:20)
  N[i]<- sum(rbinom(N[i-1],3,0.6))
```

N

```
[1]      1      2      3      7     13     23
[7]     41     79    128    236    422    789
[13]   1435   2609   4764   8594  15565  27964
[19] 50574 91053
```

From the above simulation, we see that the subcritical process dies out quickly, the critical takes a bit longer to die out, but the supercritical process grows to a large number of particles and doesn't become extinct. \square

SIMULATION 8.2. In this simulation, we generate and plot a trajectory of a branching process, in which offspring distribution is binomial with parameters $n = 3$ and $p = 0.7$. We utilize a self-referencing function in R and request to plot five generations of the process. The code is as follows.

```
library(tidyverse)

#specifying parameters gen.max<- 5
prob<- 0.7

#specifying seed
set.seed(2443534)

#simulating trajectory
level.segment<- function(gen, y, branch.num) {
  branch <- data.frame(x=c(), y=c(), xend=c(), yend=c())
  gen.remaining<- gen.max-gen-1
  if (gen.remaining < 0) return(branch)

  if (branch.num > 0) {
    branch<- rbind(branch, data.frame(x=gen, y=y, xend=gen+1,
    yend=y), level.segment(gen=gen+1, y=y, branch.num=rbinom(1,
    3, prob)))
  }

  if (branch.num > 1) {
    branch<- rbind(branch, data.frame(x=gen, y=y, xend=gen+1,
    yend=y+ 3^gen.remaining), level.segment(gen=gen+1,
    y=y+3^gen.remaining, branch.num=rbinom(1, 3, prob)))
  }
}
```

```

if (branch.num > 2) {
  branch<- rbind(branch, data.frame(x=gen, y=y, xend=gen+1,
  yend=y-3^gen.remaining), level.segment(gen=gen+1,
  y=y-3^gen.remaining, branch.num=rbinom(1, 3, prob)))
}

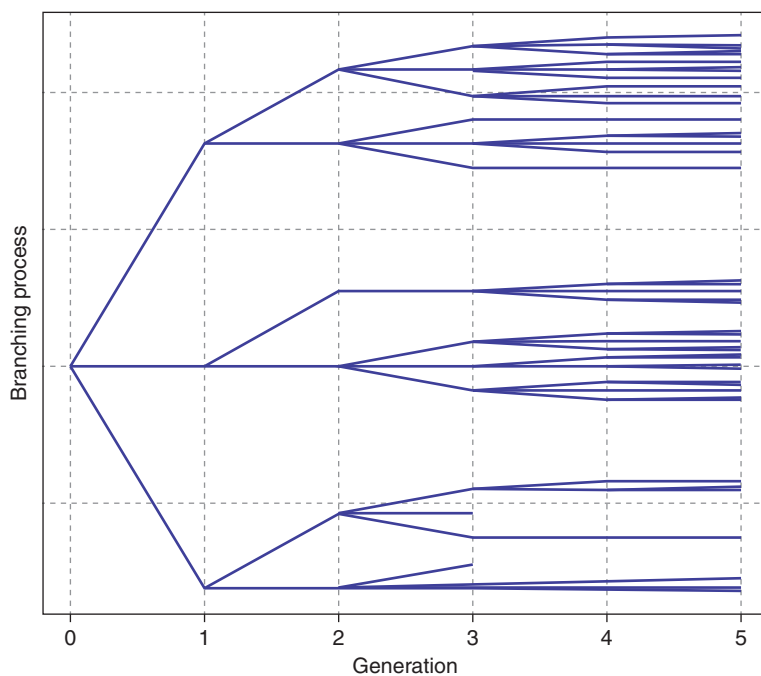
branch
}

bp<- level.segment(0, 0, rbinom(1, 3, prob))

#plotting trajectory
plot(bp[,1], bp[,2], type="n", yaxt="n", xlim=c(0,5),
ylim=c(range(bp)), xlab="Generation", ylab="Branching
process", panel.first=grid())

segments(bp[,1], bp[,2], bp[,3], bp[,4], lwd=2, col="blue")

```



8.3 Applications of Branching Process

APPLICATION 8.1. One of the research interests of cultural anthropologists is that of a long-term history of population dynamics. Suppose a certain geographic area was initially populated by 25 families with 25 women of child-bearing age. In modern days, when a census of that area was taken, there were 19,856 women of child-bearing age. Ten percent of them had one daughter, 20% had two daughters, 60% had three daughters, and the others didn't have any daughters.

(a) From these data, we can estimate the mean size of female offspring. We have $\hat{\mu} = (0)(0.1) + (1)(0.1) + (2)(0.2) + (3)(0.6) = 2.3$. This is a supercritical process. The extinction of each family in this population is not going to happen for sure. In fact, the estimated probability of extinction of each family $\hat{\pi}_0$ solves $\hat{\pi}_0 = 0.1 + 0.1\hat{\pi}_0 + 0.2\hat{\pi}_0^2 + 0.6\hat{\pi}_0^3$, or, equivalently, $6\hat{\pi}_0^3 + 8\hat{\pi}_0 - 1 = 0$. The solution is $\hat{\pi}_0 = \frac{\sqrt[3]{22-4}}{6} = 0.115069$. The estimated probability that all the 25 families become extinct is $\hat{\pi}_0^{25}$ which is very close to zero. However, the probability that at least one of the 25 families becomes extinct is computed as $1 - (1 - 0.115069)^{25} = 0.952931$.

(b) From the data, we can also assess how long ago the initial settlement took place. We know that on average, the size of the n th generation is $(25)(\hat{\mu}^n) = (25)(2.3)^n$. Therefore, $(25)(2.3)^{\hat{n}} = 19,856$. From here, $\hat{n} = \ln(19,856/25)/\ln(2.3) = 8.0169$ or about eight generations. Assuming that it takes around 25 years for a generation to mature, we can say that the settlement was established about $(8)(25) = 200$ years ago. \square

APPLICATION 8.2. In epidemiology, the most basic model of the spread of an infectious disease is a branching process. An initially infected individual will either recover, for instance, with probability 0.1 without infecting others, or will infect a zero-truncated Poisson(λ) random number of individuals where $\lambda = 2.4$, say.

(a) To find the mean of the n th generation of infected individuals, we note that the probability mass function for the offspring is $p(0) = 0.1$ and $p(n) = (0.9)\frac{\lambda^n}{n!} \frac{e^{-\lambda}}{1 - e^{-\lambda}} = (0.9)\frac{(2.4)^n}{n!} \frac{e^{-2.4}}{1 - e^{-2.4}}$, $n = 1, 2, 3, \dots$. The mean of this distribution is $\mu = (0.9)\frac{\lambda}{1 - e^{-\lambda}} = \frac{(0.9)(2.4)}{1 - e^{-2.4}} = 2.375501$. The

variance is $\sigma^2 = (0.9) \sum_{n=1}^{\infty} n^2 \frac{\lambda^n}{n!} \frac{e^{-\lambda}}{1 - e^{-\lambda}} - \mu^2 = \frac{0.9}{1 - e^{-\lambda}} (\lambda + \lambda^2) - \mu^2 = \frac{0.9}{1 - e^{-2.4}} (2.4 + (2.4)^2) - (2.375501)^2 = 2.433697$.

(b) The expected number of infected individuals in the n th generation is $\mu^n = (2.375501)^n$, and the standard deviation is

$$\begin{aligned} \sqrt{\sigma^2 \mu^{n-1} \left(\frac{1 - \mu^n}{1 - \mu} \right)} &= \sqrt{(2.433697)(2.375501)^{n-1} \left(\frac{1 - (2.375501)^n}{1 - 2.375501} \right)} \\ &= \sqrt{(1.769317)(2.375501)^{n-1} ((2.375501)^n - 1)}. \end{aligned}$$

(c) Since $\mu > 1$, this is a supercritical process. The probability π_0 that the infection stops spreading is the smallest positive solution of the equation

$$\pi_0 = (0.1)(\pi_0)^0 + \sum_{n=1}^{\infty} \pi_0^n (0.9) \frac{(2.4)^n}{n!} \frac{e^{-2.4}}{1 - e^{-2.4}},$$

or

$$\pi_0 = 0.1 + \frac{0.9}{e^{2.4} - 1} \sum_{n=1}^{\infty} \frac{(2.4\pi_0)^n}{n!} = 0.1 + \frac{0.9}{e^{2.4} - 1} (e^{2.4\pi_0} - 1).$$

Solved numerically, $\pi_0 = 0.1340992$. The lines of code that produce this answer follow.

```
library(rootSolve)
equation<- function(x)
x-0.1-0.9/(exp(2.4)-1)*(exp(2.4*x)-1)

uniroot.all(equation, c(0,0.99))
```

0.1340992

□

Exercises

EXERCISE 8.1. Consider a colony of bacteria. Bacteria are known to reproduce asexually by binary fission, splitting into two identical cells. Suppose at time 0 the colony size is $X_0 = 100$ bacteria. Suppose also that at the end of a time unit, each bacterium, independently of the others, dies with probability 0.25, splits into two with probability 0.6, or continues living with probability 0.15.

- (a) Show that the colony growth can be modeled as a supercritical branching process. Find the expected size of the n th generation and its variance.
- (b) Compute the extinction probability for descendants of each bacterium.
- (c) Find the probability that descendants of at least one of ten bacteria go extinct.

EXERCISE 8.2. Consider a branching process with a sole ancestor and a $\text{Poisson}(\lambda)$ proliferation distribution.

- (a) Determine the values of λ for which this process is supercritical (critical, subcritical).
- (b) Give expressions for the mean and variance of the size of the n th generation.
- (c) Give the equation that the extinction probability solves. Plot a graph of the numeric solution as a function of $\lambda > 1$.

EXERCISE 8.3. Based on work by Alfred J. Lotka² who analyzed the data from the 1920 U.S. Census, suppose that male offspring has a *zero-adjusted geometric distribution* of the form:

$$p(0) = 0.4828 \text{ and } p(n) = (0.228292)(0.5586)^{n-1} \text{ if } n = 1, 2, 3, \dots$$

- (a) Find the expected size of male offspring and its standard deviation.
- (b) Consider a single male ancestor. Find the expected size of the n th generation of his descendants and its standard deviation.
- (c) Compute the probability of extinction.

EXERCISE 8.4. Parlaying in gambling is defined as a series of bets in which winnings are used as a stake for further bets. This process can be modeled as a branching process. Suppose a gambler starts with a stake of \$1, and can win \$1 with probability 0.3, or \$15 with probability 0.2, or \$20 with probability 0.1, or \$0 with probability 0.4.

- (a) Show that this is a supercritical process.
- (b) Find the expected winnings on the fifth bet.
- (c) Find the probability that the gambler's stake eventually turns into \$0.

EXERCISE 8.5. The spread of computer viruses is often modeled as a branching process. Assume that initially one computer is infected with a virus, and every day the virus is sent to other computers which number has a discrete uniform distribution between 0 and 3.

²Lotka, A. J. (1931a). "The extinction of families, I." *Journal of the Washington Academy of Sciences*, 21(16): 377 – 380; and Lotka, A. J. (1931b). "The extinction of families, II." *Journal of the Washington Academy of Sciences*, 21(18): 453 – 459.

- (a) Is this process subcritical, critical, or supercritical?
- (b) What are the average number and standard deviation of infected computers on day 10?
- (c) Will the spread of viruses die out with probability 1? If not, find the extinction probability.
- (d) Simulate the number of infected computers during 10 days. What is the simulated total number of infected computers?

EXERCISE 8.6. Consider a branching process that starts with a single particle in generation 1. Assume that the offspring has a distribution with the probability mass function $p(0) = 0.1$, $p(1) = 0.4$, $p(2) = 0.5$.

- (a) Generate the size of the first 20 generations of this process. What is the size of the offspring of the 19th generation? How many total particles are in this population?
- (b) Simulate and plot a sample trajectory of this process for the first six generations.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Brownian Motion

9.1 Definition of Brownian Motion

A stochastic process $\{B(t), t \geq 0\}$ is called a *standard Brownian motion*¹ (or a *Wiener process*²) if: (i) $B(0) = 0$, (ii) it has independent and stationary increments, and (iii) $B(t) \sim N(0, t)$, $t > 0$.

REMARK 9.1. Brownian motion is sometimes termed the *Bachelier process*. In his 1900 paper³ a French mathematician Louis Bachelier derived the Brownian motion as the limit of random walks.

PROPOSITION 9.1. (a) Brownian motion is everywhere continuous but nowhere differentiable.

(b) Brownian motion hits every real number infinitely many times.

PROOF: The proofs of both statements are omitted. \square

PROPOSITION 9.2. For a standard Brownian motion, the covariance between $B(s)$ and $B(t)$ is $\text{Cov}(B(s), B(t)) = \min(s, t)$, $s, t \geq 0$.

¹English botanist Robert Brown observed the movement of dust particles in liquid and described the motion in Brown, R. (1828). “A brief account of microscopical observations made in the months of June, July and August, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies.” *Philosophical Magazine, Series 2*, 4: 161 – 173.

²Norbert Wiener proposed a rigorous mathematical model for a Brownian motion in Wiener, N. (1921). “The average of an analytic functional and the Brownian movement.” *Proceedings of the National Academy of Sciences of the United States of America*, 7(10): 294 – 298.

³Bachelier, L. (1900). “Théorie de la spéculation.” *Annales Scientifiques de l’École Normale Supérieure*, 17: 21 – 86.

PROOF: Suppose $s \leq t$. Since the mean of a Brownian motion is zero and its increments are independent, we obtain

$$\begin{aligned}\mathbb{Cov}(B(s), B(t)) &= \mathbb{E}(B(s)B(t)) = \mathbb{E}[B(s)(B(t) - B(s) + B(s))] \\ &= \mathbb{E}[B(s)(B(t) - B(s))] + \mathbb{E}(B(s))^2 = \mathbb{E}(B(s))\mathbb{E}(B(t) - B(s)) + \mathbb{E}(B(s))^2 \\ &= \mathbb{E}(B(s))^2 = \mathbb{Var}(B(s)) = s = \min(s, t). \quad \square\end{aligned}$$

PROPOSITION 9.3. (RESCALING RELATION). For any real a , $B(at)$ and $\sqrt{a}B(t)$ have the same distribution, where $B(t)$ denotes a standard Brownian motion.

PROOF: The distribution of $B(at)$ is $N(0, at)$ by the definition of a Brownian motion. Also, since $B(t) \sim N(0, t)$, we have that $\sqrt{a}B(t)$ is also normal with mean $\mathbb{E}(\sqrt{a}B(t)) = \sqrt{a}\mathbb{E}(B(t)) = 0$ and variance $\mathbb{Var}(\sqrt{a}B(t)) = a\mathbb{Var}(B(t)) = at$. \square

EXAMPLE 9.1. Suppose we want to compute the conditional probability that a standard Brownian motion is below 3 at time 3, given that it is equal to 1 at time 1. We write

$$\begin{aligned}\mathbb{P}(B(3) < 3 \mid B(1) = 1) &= \mathbb{P}(B(3) - B(1) < 3 - 1 \mid B(1) = 1) \\ &= \mathbb{P}(B(3) - B(1) < 2) \text{ (by independence of increments)} \\ &= \mathbb{P}(B(2) < 2) \text{ (by stationarity of increments)} \\ &= \mathbb{P}(\sqrt{2}B(1) < 2) \text{ (by the rescaling relation)} \\ &= \mathbb{P}(B(1) < \sqrt{2}) = \Phi(\sqrt{2}) = 0.92135. \quad \square\end{aligned}$$

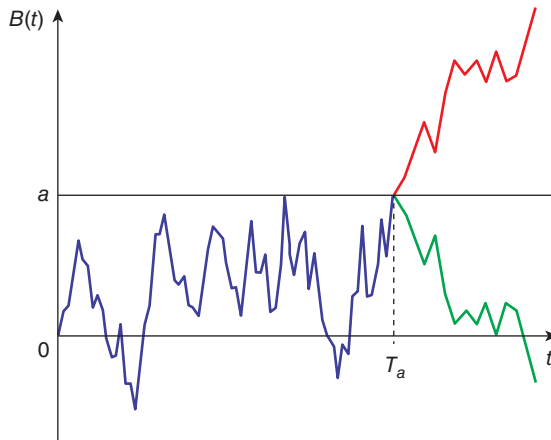
Here and later $\Phi(\cdot)$ denotes the standard normal cumulative distribution function.

PROPOSITION 9.4. (DISTRIBUTION OF HITTING TIME). Denote by T_a the first time a standard Brownian motion hits a level $a > 0$. The cumulative distribution function of T_a is $F_{T_a}(t) = 2(1 - \Phi(a/\sqrt{t}))$, $t > 0$.

PROOF: We write $\mathbb{P}(B(t) \geq a) = \mathbb{P}(B(t) \geq a \mid T_a \leq t)\mathbb{P}(T_a \leq t) + \mathbb{P}(B(t) \geq a \mid T_a > t)\mathbb{P}(T_a > t)$. If $T_a > t$, $B(t)$ hasn't reached the level a yet, and it is impossible to have $B(t) \geq a$. Thus, the second term is equal to 0.

Also, from symmetry (see the picture), the Brownian motion is as likely to go up after hitting a as down, therefore,

$$\mathbb{P}(B(t) \geq a \mid T_a \leq t) = \mathbb{P}(B(t) \leq a \mid T_a \leq t) = \frac{1}{2}.$$



Consequently, $\mathbb{P}(B(t) \geq a) = \frac{1}{2} \mathbb{P}(T_a \leq t)$, and hence, $\mathbb{P}(T_a \leq t) = 2\mathbb{P}(B(t) \geq a) = 2(1 - \Phi(a/\sqrt{t}))$. \square

REMARK 9.2. In the proof above, we used the symmetry property of a Brownian motion. Put rigorously, this property is known as the *reflection principle* and is stated as: if a path of a Brownian motion reaches a value $B(s)$ at time s , a path after time s has the same distribution as its reflection about the value $B(s)$.

EXAMPLE 9.2. The probability that a Brownian motion reaches level 1 by time 5 is $\mathbb{P}(T_1 \leq 5) = 2(1 - \Phi(1/\sqrt{5})) = 0.6547$. \square

PROPOSITION 9.5. (DISTRIBUTION OF MAXIMUM VALUE). Denote by $M(t)$ the maximum value that a Brownian motion attains on the interval $[0, t]$. The cumulative distribution function of $M(t)$ is $F_{M(t)}(a) = 2\Phi(a/\sqrt{t}) - 1$, for $a > 0$ and $t > 0$.

PROOF: Note that $M(t) > a > 0$, if and only if $T_a < t$. Therefore,

$$\begin{aligned} \mathbb{P}(M(t) \leq a) &= 1 - \mathbb{P}(M(t) > a) = 1 - \mathbb{P}(T_a < t) \\ &= 1 - 2(1 - \Phi(a/\sqrt{t})) = 2\Phi(a/\sqrt{t}) - 1. \quad \square \end{aligned}$$

EXAMPLE 9.3. The probability that a Brownian motion is below 2 everywhere on the interval $[0, 4]$ is $\mathbb{P}(M(4) < 2) = 2\Phi(2/\sqrt{4}) - 1 = 2\Phi(1) - 1 = 0.682689$. \square

9.2 Processes Derived from Brownian Motion

9.2.1 Brownian Bridge

A *Brownian bridge* is a stochastic process $\{X(t), 0 \leq t \leq 1\}$ that satisfies the following properties: (i) $X(t)$ is normally distributed, (ii) $X(0) = X(1) = 0$, (iii) $\mathbb{E}(X(t)) = 0$, (iv) $\mathbb{V}ar(X(t)) = t(1 - t)$, and (v) $\mathbb{C}ov[X(s), X(t)] = \min(s, t) - st$, $0 \leq s, t \leq 1$.

We can think of a Brownian bridge as a Brownian motion on the interval $[0, 1]$, tied at the two ends. Note that the variance is equal to 0 at both ends of the interval as it should be since the values are deterministic at those points, and increases toward the middle, reaching its maximum at $t = 1/2$.

More generally, a Brownian bridge on the interval $[0, T]$, $\{X(t), 0 \leq t \leq T\}$, is such that: (i) $X(t)$ is normally distributed, (ii) $X(0) = X(T) = 0$, (iii) $\mathbb{E}(X(t)) = 0$, (iv) $\mathbb{V}ar(X(t)) = t(1 - t/T)$, and (v) $\mathbb{C}ov[X(s), X(t)] = \min(s, t) - st/T$, $0 \leq s, t \leq T$.

PROPOSITION 9.6. Suppose that $\{B(t), t \geq 0\}$ is a standard Brownian motion, and let $X(t) = B(t) - tB(1)$, $0 \leq t \leq 1$. Then $\{X(t), 0 \leq t \leq 1\}$ is a Brownian bridge.

PROOF: $X(t)$ has a normal distribution because $B(t)$ is normally distributed. The mean of $X(t)$ is $\mathbb{E}(X(t)) = \mathbb{E}(B(t)) - t\mathbb{E}(B(1)) = 0 - (t)(0) = 0$. Now, recall that $\mathbb{C}ov(B(s), B(t)) = \mathbb{E}(B(s)B(t)) = \min(s, t)$ (see Proposition 9.2). Assuming $s \leq t$, we compute the covariance between $X(s)$ and $X(t)$ as

$$\begin{aligned} \mathbb{C}ov(X(s), X(t)) &= \mathbb{E}[B(s) - sB(1), B(t) - tB(1)] \\ &= \mathbb{E}(B(s)B(t)) - t\mathbb{E}(B(s)B(1)) - s\mathbb{E}(B(1)B(t)) + st\mathbb{E}(B(1))^2 \\ &= \min(s, t) - t\min(s, 1) - s\min(1, t) + st\mathbb{V}ar(B(1)) \\ &= s - ts - st + st = s - st = \min(s, t) - st. \quad \square \end{aligned}$$

REMARK 9.3. The proof of the above proposition can be extended (do it!) to show that $\{X(t) = B(t) - \frac{t}{T} B(T), t \geq 0\}$ is a Brownian bridge on the interval $[0, T]$.

9.2.2 Brownian Motion with Drift and Volatility

Let $\{B(t), t \geq 0\}$ denote a standard Brownian motion. A stochastic process $\{X(t) = \mu t + \sigma B(t), t \geq 0\}$ is called a Brownian motion with the *drift coefficient* μ and *volatility* (or *diffusion*) *coefficient* σ .

PROPOSITION 9.7. The distribution of $X(t)$ is normal with mean μt and variance $\sigma^2 t$. Also, the covariance between $X(s)$ and $X(t)$ is $\sigma^2 \min(s, t)$.

PROOF: The distribution of $X(t)$ is normal since $B(t)$ is normally distributed. The mean of $X(t)$ is $\mathbb{E}(X(t)) = \mu t + \sigma \mathbb{E}(B(t)) = \mu t$, and the variance is $\mathbb{V}ar(X(t)) = \mathbb{V}ar(\mu t + \sigma B(t)) = \sigma^2 \mathbb{V}ar(B(t)) = \sigma^2 t$. The covariance between $X(s)$ and $X(t)$ is $\mathbb{C}ov(X(s), X(t)) = \mathbb{E}\left((\mu s + \sigma B(s))(\mu t + \sigma B(t))\right) - \mathbb{E}(\mu s + \sigma B(s))\mathbb{E}(\mu t + \sigma B(t)) = (\mu s)(\mu t) + \sigma^2 \mathbb{E}(B(s)B(t)) - (\mu s)(\mu t) = \sigma^2 \min(s, t)$. \square

9.2.3 Geometric Brownian Motion

A stochastic process $\{Y(t) = Y(0) \exp(\mu t + \sigma B(t)), t \geq 0\}$ is called a *geometric* (or *exponential*) *Brownian motion*.

PROPOSITION 9.8. The distribution of $Y(t)$ is log-normal with the density function

$$f_{Y(t)}(y) = \frac{1}{\sqrt{2\pi\sigma^2 t} y} \exp\left(-\frac{(\ln y - \ln Y(0) - \mu t)^2}{2\sigma^2 t}\right), \quad y > 0.$$

The mean and variance are

$$\mathbb{E}(Y(t)) = Y(0) e^{\mu t + \sigma^2 t/2} \quad \text{and} \quad \mathbb{V}ar(Y(t)) = [Y(0)]^2 e^{2\mu t + \sigma^2 t} (e^{\sigma^2 t} - 1).$$

PROOF: Since $B(t) \sim N(0, t)$, the cumulative distribution function of $Y(t)$ is derived as follows:

$$\begin{aligned} F_{Y(t)}(y) &= \mathbb{P}(Y(t) \leq y) = \mathbb{P}(\mu t + \sigma B(t) \leq \ln y - \ln Y(0)) \\ &= \mathbb{P}\left(B(t) \leq \frac{\ln y - \ln Y(0) - \mu t}{\sigma}\right) = \Phi\left(\frac{\ln y - \ln Y(0) - \mu t}{\sigma \sqrt{t}}\right), \quad y > 0. \end{aligned}$$

The density is

$$f_{Y(t)}(y) = F'_{Y(t)}(y) = \frac{1}{\sqrt{2\pi\sigma^2 t} y} \exp\left(-\frac{(\ln y - \ln Y(0) - \mu t)^2}{2\sigma^2 t}\right), \quad y > 0.$$

Using the expression for the moment generating function of $B(t) \sim N(0, t)$, $\mathbb{E}(e^{\sigma B(t)}) = e^{\sigma^2 t/2}$, we get that the mean of $Y(t)$ is

$$\mathbb{E}(Y(t)) = \mathbb{E}(Y(0) e^{\mu t + \sigma B(t)}) = Y(0) e^{\mu t} \mathbb{E}(e^{\sigma B(t)}) = Y(0) e^{\mu t + \sigma^2 t/2},$$

and the variance is

$$\begin{aligned} \text{Var}(Y(t)) &= \mathbb{E}(Y(t))^2 - [\mathbb{E}(Y(t))]^2 = \mathbb{E}(Y(0) e^{\mu t + \sigma B(t)})^2 \\ &\quad - (Y(0) e^{\mu t + \sigma^2 t/2})^2 \\ &= [Y(0)]^2 e^{2\mu t} \mathbb{E}(e^{2\sigma B(t)}) - [Y(0)]^2 e^{2\mu t + \sigma^2 t} = [Y(0)]^2 e^{2\mu t + 2\sigma^2 t} \\ &\quad - [Y(0)]^2 e^{2\mu t + \sigma^2 t} \\ &= [Y(0)]^2 e^{2\mu t + \sigma^2 t} (e^{\sigma^2 t} - 1). \quad \square \end{aligned}$$

9.2.4 The Ornstein-Uhlenbeck Process

The Ornstein-Uhlenbeck process⁴ $\{X(t), t \geq 0\}$ is a stochastic process of the form

$$X(t) = X(0)e^{-\theta t} + \mu(1 - e^{-\theta t}) + \frac{\sigma}{\sqrt{2\theta}}e^{-\theta t}B(e^{2\theta t} - 1).$$

Here μ is the drift, $\sigma > 0$ is the volatility, and $\theta > 0$ is an additional parameter.

PROPOSITION 9.9. The mean of $X(t)$ is $X(0)e^{-\theta t} + \mu(1 - e^{-\theta t})$, and the variance is $\frac{\sigma^2}{2\theta}(1 - e^{-2\theta t})$.

PROOF: The mean of $X(t)$ is

$$\begin{aligned} \mathbb{E}(X(t)) &= X(0)e^{-\theta t} + \mu(1 - e^{-\theta t}) + \frac{\sigma}{\sqrt{2\theta}}e^{-\theta t} \mathbb{E}(B(e^{2\theta t} - 1)) \\ &= X(0)e^{-\theta t} + \mu(1 - e^{-\theta t}), \end{aligned}$$

⁴First appeared in Uhlenbeck, G. E. and L. S. Ornstein (1930). "On the theory of Brownian motion." *Physical Review*, 36: 823 – 841.

since the expected value of a Brownian motion (in this case, a time-transformed Brownian motion) is equal to 0. The variance of $X(t)$ is

$$\begin{aligned}\mathbb{V}ar(X(t)) &= \frac{\sigma^2}{2\theta} e^{-2\theta t} \mathbb{V}ar(B(e^{2\theta t} - 1)) = \frac{\sigma^2}{2\theta} e^{-2\theta t} (e^{2\theta t} - 1) \\ &= \frac{\sigma^2}{2\theta} (1 - e^{-2\theta t}).\end{aligned}$$

□

REMARK 9.4. Note that as t increases, the mean of the Ornstein-Uhlenbeck process tends to μ . Therefore, the drift μ is the *long-term mean*, and the process is called *mean-reverting*. The parameter θ represents the rate by which the process reverts towards the mean. In addition, the variance of this process is bounded by a constant $\sigma^2/(2\theta)$, and in the long run, approaches this constant. □

9.3 Simulations in R

SIMULATION 9.1. (ONE-DIMENSIONAL STANDARD BROWNIAN MOTION). The code below simulates three trajectories of a standard Brownian motion on the time interval that has 500 increments of size 0.01. The plot follows.

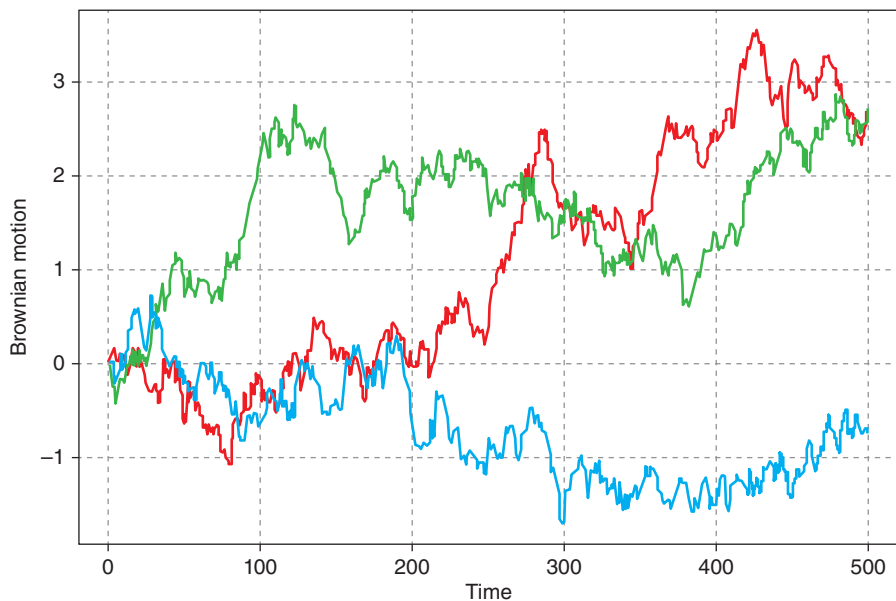
```
BM<- matrix(NA, nrow=500, ncol=3)

#specifying seed
set.seed(8221056)

#simulating trajectories
for (j in 1:3) {
  BM[1,j]<- 0

  for (i in 2:500)
    BM[i,j]<- BM[i-1,j] + sqrt(0.01)*rnorm(1)
}

#plotting trajectories
matplot(BM, type="l", lty=1, lwd=2, col=2:4,
  ylim=c(range(BM)), xlab="Time", ylab="Brownian motion",
  panel.first=grid())
```



□

SIMULATION 9.2. (TWO-DIMENSIONAL BROWNIAN MOTION). A *two-dimensional Brownian motion* is a stochastic process that keeps track of two coordinates, both of which are independent Brownian motions. The R syntax below simulates and plots one trajectory of a two-dimensional Brownian motion.

```
BM<- matrix(NA, nrow=5000, ncol=2)

#specifying seed
set.seed(34885002)

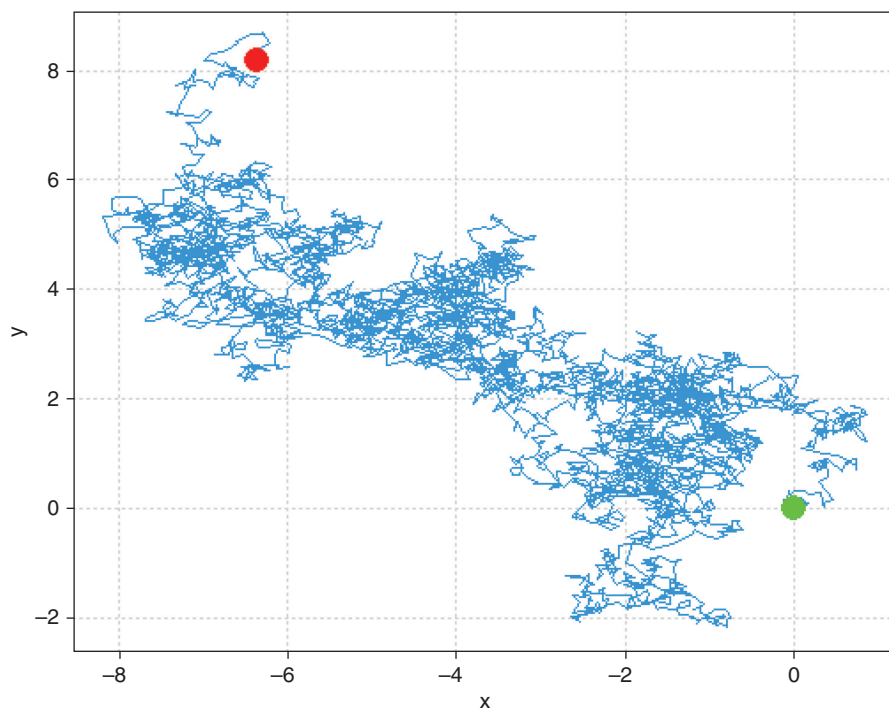
#simulating two independent Brownian motions
for (j in 1:2) {
  BM[1,j]<- 0

  for (i in 2:5000)
    BM[i,j]<- BM[i-1,j] + sqrt(0.01)*rnorm(1)
}

#plotting trajectory
plot(x=BM[,1], y=BM[,2], type="l", col=4, xlab="x", ylab="y",
     xlim=range(BM[,1]), ylim=range(BM[,2]), panel.first=grid())
```

```
#adding starting point
points(cbind(BM[1,1], BM[1,2]), pch=16, cex=2, col="green")

#adding ending point
points(cbind(BM[5000,1], BM[5000,2]), pch=16, cex=2,
col="red")
```



□

SIMULATION 9.3. (THREE-DIMENSIONAL BROWNIAN MOTION). A *three-dimensional Brownian motion* is a stochastic process that models position by three coordinates, defined by three independent Brownian motions. Below we simulate and plot a single trajectory of a three-dimensional Brownian motion.

```
nsteps<- 2000
BM<- matrix(NA, nrow=nsteps, ncol=3)

#specifying seed
set.seed(1133205)
```

```

#simulating three independent Brownian motions
for (j in 1:3) {
  BM[1,j]<- 0

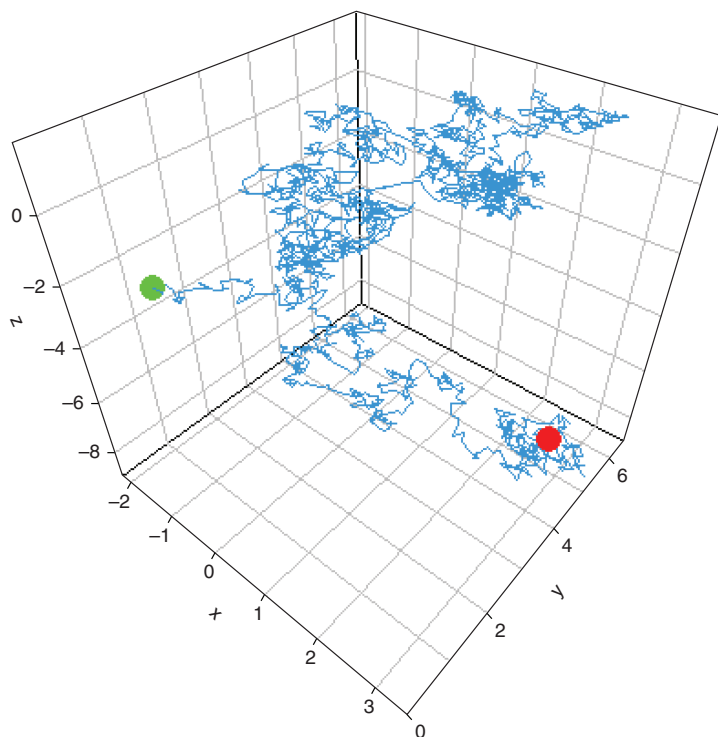
  for (i in 2:nsteps)
    BM[i,j]<- BM[i-1,j] + sqrt(0.01)*rnorm(1)
}

#plotting trajectory
library(plot3D)
lines3D(BM[,1], BM[,2], BM[,3], col=4, xlab="x", ylab="y",
        zlab="z", xlim=range(BM[,1]), ylim=range(BM[,2]),
        zlim=range(BM[,3]), bty="b2", ticktype="detailed")

#adding starting point
points3D(x=BM[1,1], y=BM[1,2], z=BM[1,3], add=TRUE, pch=16,
        cex=2, col="green")

#adding ending point
points3D(BM[nsteps,1], BM[nsteps,2], BM[nsteps,3], add=TRUE,
        pch=16, cex=2, col="red")

```



SIMULATION 9.4. (BROWNIAN BRIDGE). The following code generates three trajectories of a Brownian bridge. First we simulate three trajectories of a standard Brownian motion $\{B(t), t \in [0, 500]\}$, and then turn them into Brownian bridge trajectories by computing $\{X(t) = B(t) - \frac{t}{500}B(500), t \in [0, 500]\}$ (see Remark 9.3). The graphical output is given below.

```
#defining Brownian motion and Brownian bridge as matrices
BM<- matrix(NA, nrow=500, ncol=3)
BB<- matrix(NA, nrow=500, ncol=3)

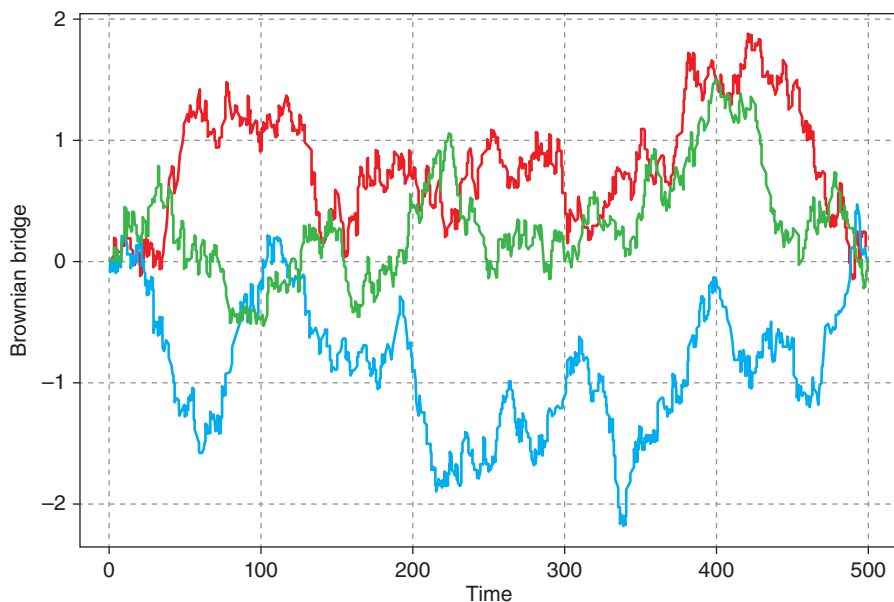
#specifying seed
set.seed(76435567)

#simulating trajectories of Brownian motion
for (j in 1:3) {
  BM[1,j]<- 0

  for (i in 2:500)
    BM[i,j]<- BM[i-1,j] + sqrt(0.01)*rnorm(1)
}

#computing trajectories of Brownian bridge
for(j in 1:3) {
  for (i in 1:500)
    BB[i,j]<- BM[i,j]-i/500*BM[500,j]
}

#plotting trajectories of Brownian bridge
matplot(BB, type="l", lty=1, lwd=2, col=2:4,
  ylim=c(range(BB)), xlab="Time", ylab="Brownian bridge",
  panel.first=grid())
```



□

SIMULATION 9.5. (BROWNIAN MOTION WITH DRIFT AND VOLATILITY). Below we generate three trajectories of a Brownian motion with drift $\mu = 1.3$ and volatility $\sigma = 0.5$. The code and plot follow.

```
#specifying parameters
mu<- 1.3
sigma<- 0.5

#defining Brownian motion as matrix
BM<- matrix(NA, nrow=500, ncol=3)

#specifying seed
set.seed(8463338)

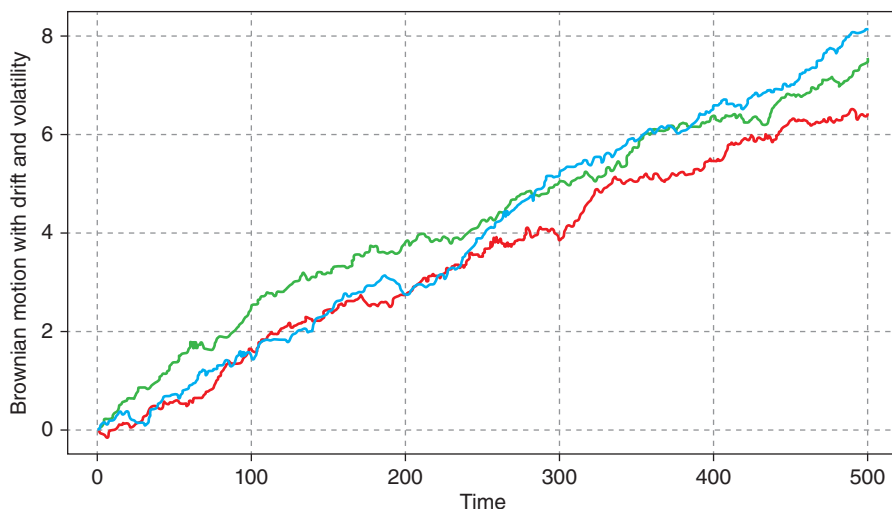
#simulating trajectories
for (j in 1:3) {
  BM[1,j]<- 0
```

```

for (i in 2:500)
  BM[i,j]<- mu*0.01+BM[i-1,j] + sigma*sqrt(0.01)*rnorm(1)
}

#plotting trajectories
matplot(BM, type="l", lty=1, lwd=2, col=2:4,
        ylim=c(range(BM)), xlab="Time", ylab="Brownian motion with
        drift and volatility", panel.first=grid())

```



□

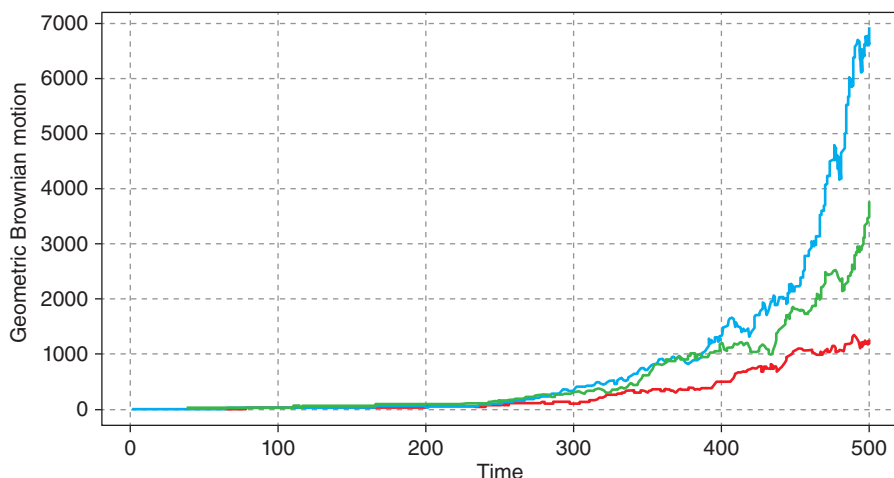
SIMULATION 9.6. (GEOMETRIC BROWNIAN MOTION). Here we simulate three trajectories of a geometric Brownian motion with the initial value $Y(0) = 2$, the drift coefficient $\mu = 1.3$, and volatility $\sigma = 0.5$. Since the trajectories of a Brownian motion with these values of drift and volatility have already been constructed in Simulation 9.5, all we need to do is to apply the exponential function to the simulated trajectories and multiply by the initial value. The code and graph follow.

```

#computing trajectories of geometric Brownian motion
GBMO<- 2
GBM<- GBMO*exp(BM)

#plotting trajectories
matplot(GBM, type="l", lty=1, lwd=2, col=2:4,
        panel.first=grid(), ylim=c(range(GBM)), xlab="Time",
        ylab="Geometric Brownian motion", panel.first=grid())

```

□

SIMULATION 9.7. (THE ORNSTEIN-UHLENBECK PROCESS). We base our simulation of a trajectory of an Ornstein-Uhlenbeck process $\{X(t), t \geq 0\}$ on the approximate recursive difference equation that it satisfies:

$$X(t + \Delta t) = X(t) + \theta(\mu - X(t))\Delta t + \sigma\sqrt{\Delta t}B(1)$$

where $\Delta t > 0$ denotes a small increment of t . We omit the proof of this formula as it involves tedious algebra.

We use this relation with time increments $\Delta t = 1$ to simulate a trajectory for the values of the parameters $X(0) = 2, \theta = 0.8, \mu = 1.6$, and $\sigma = 0.5$. The code and graph follow.

```
#specifying parameters
theta<- 0.8
mu<- 1.6
sigma<- 0.5

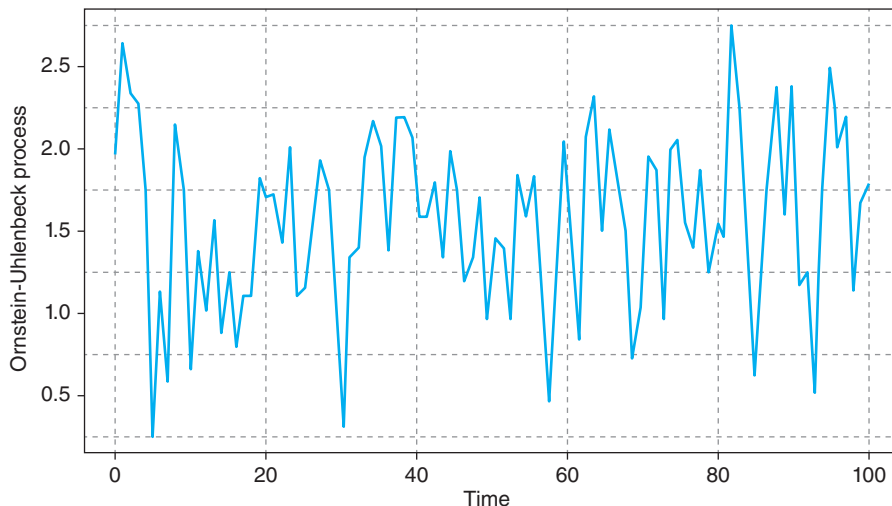
#specifying seed
set.seed(2043442)

#defining Ornstein-Uhlenbeck trajectory as vector
OU<- c()

#specifying initial value
OU[1]<- 2
```

```
#simulating trajectory
for (i in 2:100)
OU[i]<- OU[i-1]+theta*(mu-OU[i-1])+sigma*rnorm(1)

#plotting trajectory
plot(1:100, OU, type="l", lty=1, lwd=2, col=4, xlab="Time",
ylab="Ornstein-Uhlenbeck process", first.panel=grid())
```



□

9.4 Applications of Brownian Motion

APPLICATION 9.1. Animal behavior researchers use the Brownian bridge to model movements of herds as they walk on their trails during daylight time and return to their designated lodging for an overnight stay. Suppose researchers observe the movements of a herd of deer during 8 hours of daylight. The main goal of the research is to estimate the distance between the north-most and south-most points that the deer have reached. This distance approximates the diameter of the deer home range. Assuming that the unit of measurement is one-tenth of a mile, below we compute the theoretical mean diameter of the home range in miles, and then simulate 1,000 trajectories to give an empirical estimate.

(a) It can be shown (see Exercise 9.9) that the expected value of the maximum of a Brownian bridge on the interval $[0, T]$ is $\frac{1}{2}\sqrt{\frac{\pi T}{2}}$. From symmetry, it can be argued that the minimum is expected to be of the same magnitude but with a negative sign. Therefore, the expected diameter of the deer home range is $\frac{1}{2}\sqrt{\frac{\pi T}{2}} - \left(-\frac{1}{2}\sqrt{\frac{\pi T}{2}}\right) = \sqrt{\frac{\pi T}{2}}$. We will assume that T is given in minutes, and thus, $T = (8)(60) = 480$ minutes. Hence, the theoretical value of the mean diameter of the home range is $\sqrt{\frac{\pi(480)}{2}} = 27.45873$ tenths of a mile or 2.75 miles.

(b) The code below simulates 1,000 trajectories of a Brownian bridge on the time interval $[0, 480]$ with an increment step of 1, and computes the sample mean of the range for the simulated trajectories.

```
#defining Brownian motion and Brownian bridge as matrices
BM<- matrix(NA, nrow=480, ncol=1000)
BB<- matrix(NA, nrow=480, ncol=1000)

#specifying seed set.seed(6769712)

#simulating trajectories of Brownian motion
for (j in 1:1000) {
  BM[1,j]<- 0

  for (i in 2:480)
    BM[i,j]<- BM[i-1,j] + rnorm(1)
}

#computing trajectories of Brownian bridge
for(j in 1:1000){
  for (i in 1:480)
    BB[i,j]<- BM[i,j]-i/480*BM[480,j]
}

#computing ranges
range<- c()
for(j in 1:1000) {
  range[j]<- max(BB[,j])-min(BB[,j])
}

#computing sample diameter of home range
print(diameter<- mean(range))
```

26.80793

Thus, the sample diameter of the home range is $26.80793/10 = 2.68$ miles. \square

APPLICATION 9.2. A geometric Brownian motion is often used to model the behavior of a stock price over time. The data set downloaded from <https://finance.yahoo.com/quote/AMZN/history/> contains Amazon.com, Inc. daily stock prices at the closing time of stock market exchange between 01/02/2020 and 06/30/2021, a total of 377 business days. First, we plot the data.

```
stock.data<- read.csv(file="./AMZN.csv", header=TRUE,
  sep=",")

date<- as.POSIXct(stock.data$Date)
price<- stock.data$Close

#plotting stock price against date
plot(date, price, type="l", lwd=2, cex=0, col="light blue",
  xlab="Time", ylab="Stock price", first.panel=grid())
```



Now we estimate the parameters and simulate a trajectory of a geometric Brownian motion. The model for the stock price is

$$\{X(t) = X(t_1) \exp(\mu t + \sigma B(t)), t_1 \leq t \leq t_{377}\}.$$

We express the increments of the natural logarithm of the process as

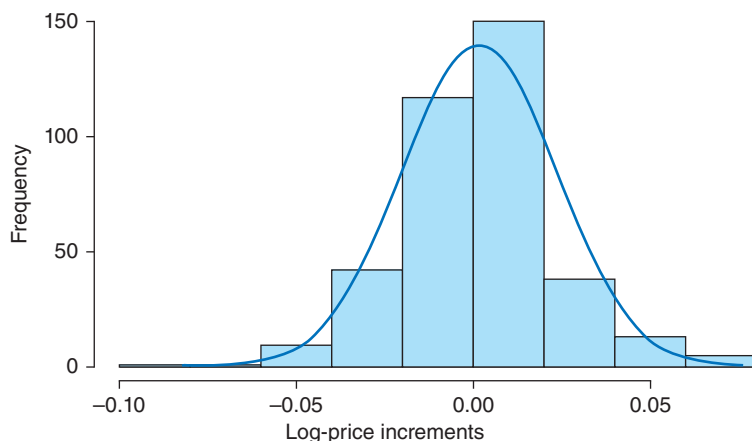
$$\ln X(t_i) - \ln X(t_{i-1}) = \ln \frac{X(t_i)}{X(t_{i-1})} = \mu(t_i - t_{i-1}) + \sigma(B(t_i) - B(t_{i-1})).$$

We take the time increments of unit length, $t_i - t_{i-1} = 1$, and argue using the stationarity of increments that the log-ratios (or log-price increments) are distributed as $\mu + \sigma B(1)$. That is, they have a normal distribution with mean μ and variance σ^2 . The code given below plots a histogram, estimates μ and σ by the sample values, simulates trajectories of the geometric Brownian motion, and plots the actual and simulated prices on the same graph.

```
#calculating increments of log-price
log.inc<- c()

price1<- price[-1]
price1.lag<- head(price, -1)
log.ratio<- log(price1/price1.lag)

#plotting histogram
library(rcompanion)
plotNormalHistogram(log.ratio, xlab="Log-price increments",
col="light blue")
```



From the histogram, a bell-shaped curve reasonably describes the density of the log-price increments, thus we conclude that the distribution can be assumed normal.

```
#estimating parameters
print(mu.hat<- mean(log.ratio))
```

```
0.001581681
```

```
print(sigma.hat<- sd(log.ratio))
```

0.02156408

```

#specifying Brownian motion as vector
BM<- c()

#specifying initial value
BM[1]<- 0

#specifying seed
set.seed(43567347)

#simulating Brownian motion with drift and volatility
for (i in 2:377)
  BM[i]<- mu.hat+BM[i-1] + sigma.hat*rnorm(1)

#computing values for geometric Brownian motion
GBM<-price[1]*exp(BM)

#plotting actual and simulated trajectories
plot(date, price, type="l", lty=1, lwd=2, col="blue",
      xlab="Time", ylab="Stock price", first.panel=grid())
lines(date, GBM, lwd=2, col="green")
legend("bottomright", c("Actual price", "Simulated price"),
      lty=1, col=c("blue", "green"))

```



APPLICATION 9.3. A geometric Brownian motion has another very famous application in the financial world. In 1997, two American economists Myron Scholes and Robert Merton were awarded the Nobel Prize in Economics for the Black-Scholes-Merton Option Pricing model. In 1973, Fischer Black and Myron Scholes⁵ published the derivation of the model and Robert Merton⁶ expanded the results. Black died in 1995, so he wasn't awarded the Nobel Prize, for it is not given posthumously.

In this application, we discuss the model and derive the final formula. First, we introduce the key concepts.

In the financial market, a *stock option* is the right to buy (or sell) a stock at a predetermined price at a fixed time in the future.

An individual can buy (or sell) the stock at the price $X(s)$ at time $s < t$, and then sell (or buy) the stock at time t for the price $X(t)$. Suppose also that the individual can buy (or sell) at time 0 a stock *option* that gives him the right to buy a stock at time t for the price K per share. How much should he pay for one share of the stock option?

Suppose we loan out \$1 today with a risk-free interest compounded continuously at the fixed rate r . Then by time t , it would grow into amount $\$e^{rt}$. From here, we conclude that a \$1 at time t is worth $\$e^{-rt}$ in today's money. The rate r is termed the *discount factor*, and the function e^{-rt} is called the *discount function*. It represents the *present value* of an amount of \$1 at time t .

We will assume that it is a fair market and there is no opportunity for an *arbitrage*. That is, there is no opportunity for a sure profit. Under this assumption, the expected return of buying (selling) one share of stock at time $s < t$ and selling (buying) it at time t is zero, which translates into the identity involving the present values of the stock prices,

$$\mathbb{E}[e^{-rt} X(t) | X(u), 0 \leq u \leq s] = e^{-rs} X(s). \quad (9.1)$$

Turning to the stock option, if the price of one share of stock at time t is below K , it is not reasonable to exercise the option. Therefore, the present value of the option is $e^{-rt}(X(t) - K)$, if $X(t) \geq K$, and 0, otherwise, which can be written as $e^{-rt}(X(t) - K)^+$.

⁵Black, F. and M. Scholes (1973). "The pricing of options and corporate liabilities." *Journal of Political Economy*, 81(3): 637 – 654.

⁶Merton, R. (1973). "Theory of rational option pricing." *Bell Journal of Economics and Management Science*, 4 (1): 141 – 183.

Let C be the price of one share of option at time zero. This is the quantity that we need to determine. In order not to create an arbitrage opportunity, we must have

$$\mathbb{E}[e^{-rt}(X(t) - K)^+ - C] = 0, \text{ or } C = e^{-rt} \mathbb{E}[(X(t) - K)^+]. \quad (9.2)$$

The Black-Scholes-Merton model assumes that $X(t)$ is a geometric Brownian motion $\{X(t) = X(0)e^{\mu t + \sigma B(t)}, t \geq 0\}$ with the drift parameter μ and volatility σ . First, we see under what condition $\{X(t), t \geq 0\}$ satisfies (9.1), and then plug this process into (9.2) to derive the final expression for C .

Using the independence and stationarity of increments of the Brownian motion and utilizing the expression of its moment generating function, we write

$$\begin{aligned} \mathbb{E}[e^{-rt} X(t) | X(u), 0 \leq u \leq s] &= e^{-rt} X(0) \mathbb{E}[e^{\mu t + \sigma B(t)} | B(u), 0 \leq u \leq s] \\ &= e^{-rt} X(0) e^{\mu t + \sigma B(s)} \mathbb{E}[e^{\sigma(B(t) - B(s))}] = e^{-rt} X(s) e^{\mu(t-s)} \mathbb{E}[e^{\sigma B(t-s)}] \\ &= e^{-rt} X(s) e^{\mu(t-s) + \frac{\sigma^2}{2}(t-s)} = e^{-rt + (\mu + \sigma^2/2)t - (\mu + \sigma^2/2)s} X(s) = e^{-rs} X(s), \end{aligned}$$

if and only if $\mu + \sigma^2/2 = r$. This is the sought-for condition on the process $\{X(t), t \geq 0\}$. We now use it in the expression (9.2). We compute

$$C = e^{-rt} \mathbb{E}[(X(t) - K)^+] = e^{-rt} \int_{-\infty}^{\infty} (X(0) e^{\mu t + \sigma \sqrt{t} z} - K)^+ \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz.$$

We want $X(0) e^{\mu t + \sigma \sqrt{t} z} - K \geq 0$, so $z \geq (\ln(K/X(0)) - \mu t)/(\sigma \sqrt{t})$. Denote by $A = (\mu t - \ln(K/X(0)))/(\sigma \sqrt{t})$. Then the lower limit of integration is $-A$. We continue

$$\begin{aligned} C &= X(0) e^{-(\mu + \sigma^2/2)t} \int_{-A}^{\infty} e^{\mu t + \sigma \sqrt{t} z} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz \\ &\quad - e^{-rt} K \int_{-A}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz \\ &= X(0) \int_{-A}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{(z - \sigma \sqrt{t})^2}{2}} dz - e^{-rt} K \int_{-A}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz \\ &= X(0) \int_{-\infty}^{A + \sigma \sqrt{t}} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz - e^{-rt} K \int_{-\infty}^A \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz \\ &= X(0) \Phi(A + \sigma \sqrt{t}) - e^{-rt} K \Phi(A). \end{aligned}$$

To work with a numeric example, suppose the current price of one share of a stock is $X(0) = \$100$. Suppose the stock price can be modeled by the

Black-Scholes-Merton model with the drift coefficient $\mu = -0.45$ and volatility $\sigma = 1.1$. We want to compute the cost of the option to buy one share of the stock at time $t = 2$ for the cost of $K = \$120$. We write

$$A = \frac{\mu t - \ln(K/X(0))}{\sigma \sqrt{t}} = \frac{(-0.45)(2) - \ln(120/100)}{(1.1)\sqrt{2}} = -0.69574,$$

$$r = \mu + \frac{\sigma^2}{2} = -0.45 + \frac{(1.1)^2}{2} = 0.155,$$

and

$$C = X(0) \Phi(A + \sigma \sqrt{t}) - e^{-rt} K \Phi(A)$$

$$= (100)\Phi(-0.69574 + (1.1)\sqrt{2}) - e^{-(0.155)(2)} (120) \Phi(-0.69574) = \$59.09. \quad \square$$

APPLICATION 9.4. As opposed to stock and option prices that can rise indefinitely, interest rates and commodity prices move in a limited range. If their values are high, the demand drops, and consequently, the values drop. Likewise, if the values are low, demand increases, and eventually, the values increase. This characteristic is called a *reversion to a long-run mean*.

The Ornstein-Uhlenbeck (OU) process is a good mathematical model that captures this mean-reversion property. Below we fit the parameters of the OU process to a publicly available data set on daily natural gas prices between 1/4/2010 and 8/11/2020 (downloaded from *kaggle.com*). Recall that the OU process solves the difference equation

$$X(t + \Delta t) = X(t) + \theta(\mu - X(t))\Delta t + \sigma\sqrt{\Delta t}B(1).$$

Using $\Delta t = 1$, we can rewrite this equation as

$$X(t + 1) - X(t) = \theta\mu - \theta X(t) + \sigma B(1),$$

and note that this has the form of a linear regression of $X(t + 1) - X(t)$ on $X(t)$. Denoting by a and b the fitted intercept and slope, respectively, we can write $\hat{\theta} = -b$ and $\hat{\mu} = a/\hat{\theta} = -a/b$. The volatility σ is estimated as the sample standard deviation of the error term. The code below estimates the parameters of the OU model and plots the observed and simulated trajectories.

```
gasprice.data<- read.csv(file="./gaspricedata.csv",
header=TRUE, sep=",")
```

```

#estimating parameters
inc<- gasprice.data$Price[-1]-head(gasprice.data$Price,-1)
fit<- glm(inc ~ head(gasprice.data$Price,-1))
theta.hat<- -fit$coefficients[2]
mu.hat<- fit$coefficients[1]/theta.hat
sigma.hat<- sigma(fit)

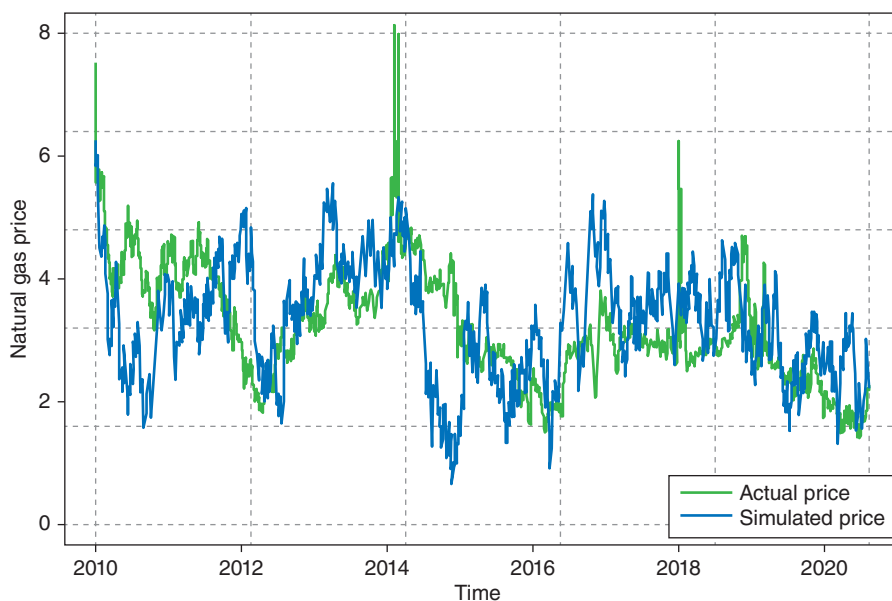
#specifying seed
set.seed(9467108)

#simulating OU process
OU<- c()
OU[1]<- gasprice.data$Price[1]

for (i in 2:length(gasprice.data$Date))
OU[i]<- OU[i-1]+theta.hat*(mu.hat-OU[i-1])+sigma.hat*rnorm(1)

#plotting trajectories
plot(as.Date(gasprice.data$Date), gasprice.data$Price,
type="l", lty=1, lwd=2, col=3, ylim=c(0,8), xlab="Time",
ylab="Natural gas price", first.panel=grid())
lines(as.Date(gasprice.data$Date), OU, lwd=2, col=4)
legend("bottomright", c("Actual price", "Simulated price"),
lty=1, col=3:4)

```

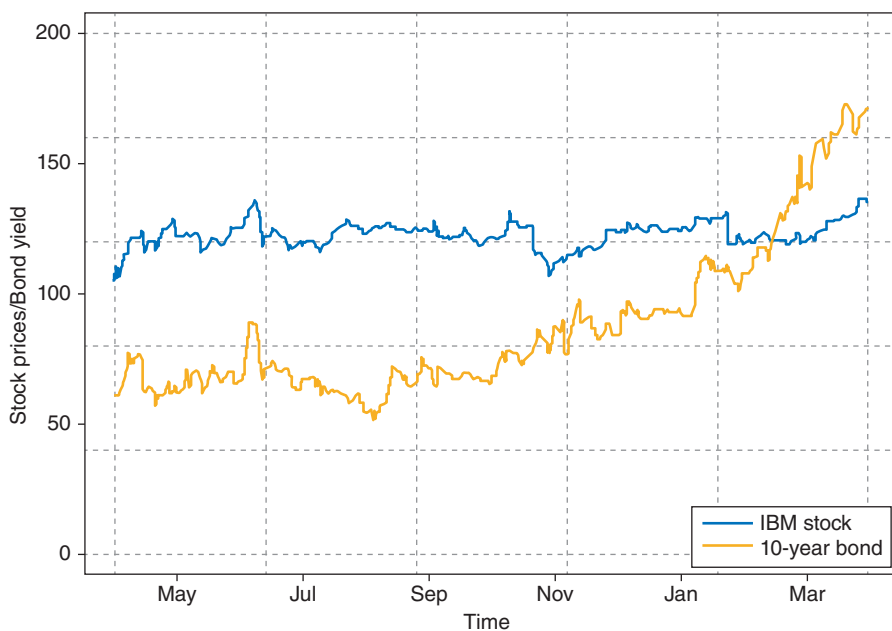


APPLICATION 9.5. Investors are interested in estimating correlation between various financial investments (for example, stock prices, stock option prices, bond yields, and commodity prices). To illustrate the concept of correlated Brownian motions, below we plot IBM stock prices at the closing of the stock market between 4/1/2020 and 3/30/2021, and U.S. 10-year treasury bond yields for the same time period. The data were downloaded from <https://www.investing.com>. The yields were rescaled by a multiplicative factor of 100 to plot comparable values.

```
data<- read.csv(file="./stock_bonds.csv", header=TRUE,
sep=",")

time<- as.Date(data$date)
IBM<- data$stock_price
bond<- data$bond_yield*100

#plotting the trajectories
plot(time, IBM, type="l", lty=1, lwd=2, col="blue",
ylim=c(0,200), xlab="Time", ylab="Stock price / Bond yield",
first.panel=grid())
lines(time, bond, lwd=2, col="orange")
legend("bottomright", c("IBM stock", "10-year bond"), lty=1,
lwd=3, col=c("blue", "orange"))
```



From the graph, the two curves exhibit somewhat similar behavior. To estimate the correlation coefficient between these two processes, we model them as correlated Brownian motions with the correlation coefficient ρ . To remove the time dependence, we resort to considering fixed-time increments where time steps are of size 1. The new processes are still correlated Brownian motions with the same correlation coefficient ρ (see Exercise 9.15 for proof). The sample Pearson correlation coefficient, computed on the increments, is the maximum-likelihood estimator of ρ . In our setting, the estimated correlation coefficient between IBM stock prices and 10-year treasury bond yields is about 0.36. The code and output follow.

```
#computing increments
IBM.diff<- IBM[-1]-head(IBM,-1)
bond.diff<- bond[-1]-head(bond,-1)

#estimating correlation coefficient
cor(IBM.diff, bond.diff)
```

0.3612484

□

Exercises

EXERCISE 9.1. Let $\{B(t), t \geq 0\}$ be a standard Brownian motion. Show that the correlation between $B(s)$ and $B(t)$ is

$$\rho(B(s), B(t)) = \sqrt{\frac{\min(s, t)}{\max(s, t)}}, \quad s, t \geq 0.$$

EXERCISE 9.2. Show that the following processes are standard Brownian motions.

- (a) $X(t) = tB(1/t)$ if $t > 0$, and 0 if $t = 0$, where $B(t)$ is a standard Brownian motion.
- (b) $Y(t) = \alpha B_1(t) + \sqrt{1 - \alpha^2} B_2(t)$, $t \geq 0$, where $B_1(t)$ and $B_2(t)$ are independent standard Brownian motions, and $0 < \alpha < 1$.

EXERCISE 9.3. Let $\{B(t), t \geq 0\}$ be a standard Brownian motion. Find the probability that

- (a) $0 < B(1) < 1$ and $1 < B(3) - B(1) < 3$.
- (b) $0 < B(1) < 1$ and $1 < B(2) < 3$. Calculate numeric value in R.
- (c) $0 < B(1) < 1$ and $0 < B(2) < \infty$.

EXERCISE 9.4. Let $\{B(t), t \geq 0\}$ be a standard Brownian motion. Suppose $0 \leq s < t$. Show that the distribution of $B(s) + B(t)$ is normal with mean 0 and variance $3s + t$.

EXERCISE 9.5. Let $\{B(t), t \geq 0\}$ denote a standard Brownian motion, and let $M(t) = \max_{0 \leq s \leq t} B(s)$. By Proposition 9.5, the cumulative distribution function of $M(t)$ is $F_{M(t)}(x) = 2\Phi(x/\sqrt{t}) - 1$, $x \geq 0$.

(a) Prove that $M(t)$ has the same distribution as $|B(t)|$, the absolute value of $B(t)$.

(b) Show that the expected value of $M(t)$ is $\mathbb{E}(M(t)) = \sqrt{\frac{2t}{\pi}}$.

(c) Compute the mean of the maximum on the interval $[0, 5]$. Simulate 1,000 trajectories of a standard Brownian motion on this interval and find an empirical estimate of the mean of the maximum.

EXERCISE 9.6. Let $\{B(t), t \geq 0\}$ denote a standard Brownian motion. Show that

- (a) $\{X(t) = -B(t), t \geq 0\}$ is also a standard Brownian motion.
- (b) $\mathbb{P}(\min_{0 \leq s \leq t} B(s) \leq x) = 2\Phi(x/\sqrt{t})$ where $x \leq 0$.
- (c) Find the probability that the minimum of a standard Brownian motion is below -3 on the interval $[0, 5]$.
- (d) Generate 1,000 trajectories of a standard Brownian motion on the interval $[0, 5]$ and find the sample probability that the minimum falls below -3.

EXERCISE 9.7. (a) Let $\{B(t), t \geq 0\}$ be a standard Brownian motion. Show that

$$X(t) = \begin{cases} (1-t)B\left(\frac{t}{1-t}\right), & \text{if } 0 \leq t < 1, \\ 0, & \text{if } t = 1 \end{cases}$$

is a Brownian bridge.

(b) Let $\{X(t), 0 \leq t \leq 1\}$ be a Brownian bridge. Show that

$$B(t) = (1+t)X\left(\frac{t}{1+t}\right), \quad t \geq 0,$$

is a standard Brownian motion.

(c) Suppose $\{X(t), 0 \leq t \leq 1\}$ is a Brownian bridge and Z is a standard normal random variable independent of the Brownian bridge. Show that $B(t) = X(t) + tZ$ is a standard Brownian motion on $[0, 1]$.

EXERCISE 9.8. Let $\{B(t), t \geq 0\}$ be a standard Brownian motion.

(a) Show that for $0 \leq s < t$, the conditional distribution of $B(s)$, given $B(t)$, is normal with mean $\frac{s}{t}B(t)$ and variance $\frac{s}{t}(t-s)$.

(b) Let $B(t) = 0$. Argue that the process in part (a) is a Brownian bridge on the interval $[0, t]$. Note: This gives us another way to define a Brownian bridge, as a Brownian motion conditioned on the value at the endpoint.

EXERCISE 9.9. Let $\{B(t), 0 \leq t \leq T\}$ be a standard Brownian motion on the interval $[0, T]$, and let $M(T)$ denote the maximum of this process.

(a) Use the reflection principle to argue that $\mathbb{P}(M(T) \geq a, B(T) \leq x) = \mathbb{P}(B(T) \geq 2a - x), a > 0, x \leq a$.

(b) Show that the conditional density of $M(T)$ given that $B(T) = x$ has the form

$$f_{M(T)|B(T)}(a|x) = \frac{2(2a-x)}{T} e^{-\frac{2a(a-x)}{T}}, \quad a > 0, x \leq a.$$

(c) Denote by $M_{BB}(T)$ the maximum of a Brownian bridge on the interval $[0, T]$. By the result of Exercise 9.8, the maximum of a Brownian bridge is the maximum of a Brownian motion conditioned on the value at time T . Use the formula derived in part (b) to show that the density of $M_{BB}(T)$ is

$$f_{M_{BB}(T)}(a) = \frac{4a}{T} e^{-\frac{2a^2}{T}}, \quad a > 0.$$

(d) Show that the expected value of $M_{BB}(T)$ is $\frac{1}{2}\sqrt{\frac{\pi T}{2}}$.

EXERCISE 9.10. A herd of bison graze on a field and return to the water source once a day.

(a) Model the daily movement of the herd as a two-dimensional Brownian bridge, where both coordinates are independent Brownian bridges. Use minutes as time units. Plot a simulated trajectory.

(b) Suppose the home range of the herd is rectangular in shape and the linear distance unit is one-tenth of a mile. Find its expected area in square miles. Hint: Use the formula for the mean value of the diameter of a one-dimensional home range derived in Application 9.1.

(c) Simulate 1,000 trajectories of the daily movement of the herd and produce an empirical estimate of the area covered, assuming a rectangular shape of the home range. How does it compare to the theoretical value from part (b)?

EXERCISE 9.11. Ornithologists have collected data on bird population size in a bird viewing preserve for 5 years (60 months). The data are given in the table below.

Month	Size	Month	Size	Month	Size	Month	Size
1	10	16	232	31	472	46	888
2	14	17	276	32	510	47	927
3	47	18	331	33	510	48	898
4	50	19	346	34	523	49	949
5	60	20	348	35	594	50	979
6	91	21	369	36	565	51	994
7	118	22	405	37	634	52	1025
8	166	23	399	38	631	53	1003
9	119	24	410	39	671	54	962
10	123	25	460	40	744	55	968
11	109	26	400	41	782	56	991
12	160	27	432	42	786	57	982
13	168	28	458	43	773	58	959
14	216	29	460	44	784	59	973
15	240	30	478	45	842	60	974

These data can be modeled as a Brownian motion with drift and volatility.

- Plot the data.
- Compute the increments and construct a histogram. Are the increments normally distributed?
- Estimate the drift and volatility coefficients.
- Simulate a Brownian motion with the estimated parameters. Overlay the actual and simulated data on the same plot.

EXERCISE 9.12. The United States Environmental Protection Agency monitors an Air Quality Index (AQI) in a certain region. The data given below contain the values of AQI for 100 consecutive days for that region.

Day	AQI	Day	AQI	Day	AQI	Day	AQI	Day	AQI
1	108	21	48	41	40	61	31	81	24
2	98	22	43	42	35	62	28	82	22
3	86	23	43	43	31	63	25	83	22
4	78	24	39	44	30	64	24	84	20
5	85	25	49	45	26	65	31	85	23
6	77	26	46	46	24	66	29	86	21
7	70	27	49	47	27	67	22	87	20
8	63	28	44	48	25	68	20	88	27
9	58	29	46	49	33	69	28	89	35
10	53	30	42	50	32	70	25	90	32
11	44	31	44	51	39	71	23	91	36
12	40	32	40	52	35	72	22	92	33
13	47	33	48	53	35	73	28	93	25
14	48	34	43	54	32	74	25	94	23
15	46	35	43	55	27	75	21	95	20
16	50	36	39	56	20	76	14	96	18
17	47	37	40	57	24	77	22	97	13
18	42	38	36	58	31	78	27	98	12
19	38	39	43	59	29	79	34	99	19
20	46	40	47	60	26	80	31	100	17

- Plot the values of AQI against time (in days). Argue that the data may be modeled via a geometric Brownian motion.
- Estimate the parameters of the geometric Brownian motion model.
- Plot the actual and simulated values in the same coordinate system.

EXERCISE 9.13. The current price of a stock is \$150. Suppose that the price of the stock changes according to a geometric Brownian motion with the drift coefficient $\mu = -0.4$ and variance $\sigma^2 = 0.76$. Use the Black-Scholes-Merton option pricing model to calculate the cost of an option to buy the stock at time $t = 7$ for a cost of \$120.

EXERCISE 9.14. Foreign currency exchange rates can be modeled well with an Ornstein-Uhlenbeck (OU) process.

- Explain in simple words why the mean-reverting property and bounded variance are expected in this setting.
- The data file “Foreign_Exchange_Rates.csv” (<https://www.kaggle.com/brunotly/foreign-exchange-rates-per-dollar-20002019>) contains daily exchange rates for some currencies (as a ratio to US dollars) between 1/3/2000 and 12/31/2019. Select a currency and estimate the parameters of the OU process. Plot actual and simulated trajectories on the same graph.

EXERCISE 9.15. Let $\{B_1(t), t \geq 0\}$ and $\{B_2(t), t \geq 0\}$ be two independent standard Brownian motions. Consider a new process $\{B_3(t), t \geq 0\}$ formed as a linear combination of these two processes: $B_3(t) = \rho B_1(t) + \sqrt{1 - \rho^2} B_2(t)$ for some fixed ρ , $-1 < \rho < 1$.

- (a) Show that $\{B_3(t), t \geq 0\}$ is a standard Brownian motion.
- (b) Show that $\{B_1(t), t \geq 0\}$ and $\{B_3(t), t \geq 0\}$ are correlated with the correlation coefficient ρ .
- (c) Show that for some fixed $s < t$, the increments $B_1(t) - B_1(s)$ and $B_3(t) - B_3(s)$ are correlated with the correlation coefficient ρ .
- (d) Download historical data from *investing.com* website for financial investments of your choice, plot the two processes, and estimate the correlation coefficient between them.

Recommended Books

- [1] Dobrow, R. P. *Introduction to Stochastic Processes with R*, Wiley, 2016.
- [2] Ross, S. M. *Introduction to Probability Models*, Academic Press, 12th edition, 2019.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

--- *List of Notations* ---

$B(t)$, 153
 $M(t)$, 155
 $N(t)$, 61, 81, 105, 117
 $P_n(t)$, 130
 P_{ij} , 1, 2
 P_{ij}^n , 3
 $P_{n,n+1}$, 129
 $P_{n,n-1}$, 129
 S , 1, 43, 129
 S_n , 62
 T , 1, 156
 T_a , 154
 T_n , 61
 $X(t)$, 1, 105, 129, 156, 158
 X_n , 1, 43, 141
 Y_i , 105
 Z_i , 46, 141
 Δt , 82
 Λ , 117
 $\Lambda(t)$, 81
 λ , 61

$\lambda(t)$, 81
 λ_n , 129
 \mathbf{P} , 2
 $\mathbf{P}^{(n)}$, 3
 \mathbf{P}_j , 51
 μ , 141, 143, 157
 μ_n , 129
 π_0 , 143
 π_j , 7
 σ , 157
 σ^2 , 141
 d , 6, 44
 $i \leftrightarrow j$, 5
 $i \rightarrow j$, 5
 p , 43
 p_i^0 , 4
 p_j^n , 4
 p_k , 141
 $p_{i,i+1}$, 43
 $p_{i,i-1}$, 43
 t , 1, 61



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

Index

n -step transition probability, 3
 n -step transition probability
matrix, 3

A

Absorbing Markov chain, 6
Absorbing state, 6
Accessible state, 5
Aperiodic state, 6
Arbitrage, 172
Asymmetric random walk, 43

B

Bachelier process, 153; *see also*
Brownian motion
Balance equations, 135
Barrier, 44
Bienaymé-Galton-Watson
process, 141; *see also*
Branching process
Birth-and-death process, 129
Birth-and-death process with
immigration and
emigration, 138
Black-Scholes-Merton
model, 172
Border, 44
Bordered random walk, 43
Boundary, 44
Branching process, 141
critical, 143
generation, 141
initial ancestor, 142
offspring, 141
probability of extinction,
143

subcritical, 143
supercritical, 143
Brownian bridge, 156
Brownian motion
diffusion coefficient, 157
drift coefficient, 157
geometric, 157
hitting time, 154
maximum value, 155
one-dimensional, 159
reflection principle, 155
rescaling relation, 154
standard, 153
three-dimensional, 161
two-dimensional, 160
volatility coefficient, 157
with drift and volatility, 157

C

Chapman-Kolmogorov equations,
3
Class, 5
Class property, 5
Collective risk model, 111
Compound Poisson process, 105
Conditional Poisson process, 117
Continuous-time Markov chain, 63
Counting process, 61
Critical branching process, 143

D

Diffusion coefficient, 157; *see also*
Volatility coefficient
Discount factor, 172
Discount function, 172

Discrete process, [1](#); *see also*
[Discrete-time process](#)
 Discrete-time Markov chain, [1](#)
 Discrete-time process, [1](#)
 Drift coefficient, [157](#)

E

Equilibrium equations, [135](#); *see also*
[Balance equations](#)
 Ergodic chain, [8](#)
 Event time, [62](#)
 Exponential Brownian motion,
[157](#); *see also* [Geometric](#)
[Brownian motion](#)

F

Finite-grid random walk, [43](#)
 Finite-state random walk, [43](#), [44](#)

G

Gambler's ruin problem, [51](#)
 Gamma-Poisson mixture
 distribution, [125](#)
 Geometric Brownian motion, [157](#)

H

Homogeneous Poisson process, [61](#)

I

Independent increments, [61](#)
 Index set, [1](#)
 Infinite random walk, [43](#)
 Integrated intensity rate function,
[81](#)
 Intensity function, [81](#); *see also*
[Intensity rate](#)
 Intensity rate, [81](#)
 power-law, [97](#)
 Interarrival time, [61](#)
 Invariant measure, [9](#)
 Irreducible chain, [5](#)

K

Kolmogorov forward equations,
[130](#)

L

Limiting distribution, [8](#);
see also [Limiting](#)
[probability](#)
 Limiting probability, [7](#), [135](#)
 Linear birth-and-death process,
[131](#)
 Linear death process, [138](#)
 Long-term mean, [159](#)

M

M/M/1 queue, [135](#)
 Markov chain
 n -step transition probability,
[3](#)
 n -step transition probability
 matrix, [3](#)
 absorbing, [6](#)
 absorbing state, [6](#)
 accessible state, [5](#)
 Chapman-Kolmogorov
 equations, [3](#)
 class, [5](#)
 class property, [5](#)
 communication property, [5](#)
 discrete-time, [1](#)
 ergodic, [8](#)
 ever enters a certain state, [5](#)
 invariant measure, [9](#)
 irreducible, [5](#)
 limiting distribution, [8](#)
 limiting probability, [7](#)
 Markov property, [2](#)
 non-ergodic, [9](#)
 one-step transition
 probability, [2](#)
 one-step transition
 probability matrix, [2](#)
 period, [6](#)
 recurrent class, [6](#)
 recurrent state, [5](#)
 states communicate, [5](#)
 stationary distribution, [7](#)
 steady-state distribution, [7](#)
 transient class, [6](#)

- transient state, 5
- unconditional distribution, 4
- Markov property, 2
- Markovian property, 2; *see also*
 - Markov property
- Mean value function, 81; *see also*
 - Integrated intensity rate function
- Mean-reverting process, 159, 174
- Memoryless property, 62
- Mixed Poisson process, 117; *see also* Conditional Poisson process

- N**
- Non-ergodic chain, 9
- Nonhomogeneous Poisson process, 81

- O**
- One-dimensional Brownian motion, 159
- One-dimensional random walk, 43
- One-step transition probability, 2
- One-step transition probability matrix, 2
- Ornstein-Uhlenbeck process, 158

- P**
- Period, 6
- Periodicity, 6
- Poisson process, 61
 - homogeneous, 61
 - nonhomogeneous, 81
- Power-law intensity rate, 97
- Present value, 172
- Probability of a return, 46
- Probability of extinction, 143
- Pure birth process, 131

- R**
- Random walk, 43
 - on a graph, 44
 - asymmetric, 43
 - barrier, 44
 - border, 44
 - bordered, 43
 - boundary, 44
 - finite grid, 43
 - finite-state, 44
 - infinite, 43
 - one-dimensional, 43
 - probability of a return, 46
 - recurrent, 46
 - simple, 43
 - symmetric, 43
 - symmetric in d dimensions, 44
 - symmetric in two dimensions, 43
 - three-dimensional, 43
 - transient, 46
 - two-dimensional, 43
 - with delay, 44
 - with loop, 44
- Recurrent class, 6
- Recurrent random walk, 46
- Recurrent state, 5
- Reflection principle, 155
- Repair rate, 97; *see also*
 - Power-law intensity rate

- S**
- Simple random walk, 43
- Splitted Poisson processes, 65; *see also* Thinned Poisson processes
- Standard Brownian motion, 153
- State, 1
- State space, 1
- States communicate, 5
- Stationary distribution, 7; *see also*
 - Limiting distribution
- Stationary increments, 61
- Stationary Poisson process, 61; *see also* Homogeneous Poisson process
- Steady-state distribution, 7; *see also*
 - Limiting probability

Steady-state probability, [135](#); *see also* [Limiting probability](#)

Stochastic process, [1](#)

time, [1](#)

continuous-time, [1](#)

discrete-time, [1](#)

index set, [1](#)

state, [1](#)

state space, [1](#)

Stock option, [172](#)

Subcritical branching process,

[143](#)

Supercritical branching process,

[143](#)

Superposition of Poisson

processes, [65](#)

Symmetric random walk,

[43](#)

Symmetric random walk in two

dimensions, [43](#)

T

Thinned Poisson processes,

[65](#)

Three-dimensional Brownian

motion, [161](#)

Three-dimensional random

walk, [43](#)

Time, [1](#)

Time-dependent Poisson process,

[81](#); *see also*

[Nonhomogeneous](#)

[Poisson process](#)

Transient class, [6](#)

Transient random walk, [46](#)

Transient state, [5](#)

Two-dimensional Brownian

motion, [160](#)

Two-dimensional random walk, [43](#)

U

Unconditional distribution, [4](#)

V

Volatility coefficient, [157](#)

W

Waiting time, [62](#); *see also* [Event](#)

[time](#)

Wiener process, [153](#); *see also*

[Brownian motion](#)

Y

Yule process, [137](#)

Z

Zero-adjusted geometric
distribution, [150](#)

SOLUTIONS MANUAL

FOR

Korosteleva, O. (2022). *Stochastic Processes with R: An Introduction*

By

OLGA KOROSTELEVA

Department of Mathematics and Statistics

California State University, Long Beach

TABLE OF CONTENTS

CHAPTER 1	3
CHAPTER 2	31
CHAPTER 3	41
CHAPTER 4	48
CHAPTER 5	60
CHAPTER 6	67
CHAPTER 7	74
CHAPTER 8	81
CHAPTER 9	87

CHAPTER 1

EXERCISE 1.1. For a Markov chain with a one-step transition probability matrix $\begin{bmatrix} 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \\ 0.8 & 0.1 & 0.1 \end{bmatrix}$ we compute:

- (a) $P(X_3 = 2 | X_0 = 1, X_1 = 2, X_2 = 3) = P(X_3 = 2 | X_2 = 3)$ (by the Markov property)
 $= P_{32} = 0.1.$
- (b) $P(X_4 = 3 | X_0 = 2, X_3 = 1) = P(X_4 = 3 | X_3 = 1)$ (by the Markov property)
 $= P_{13} = 0.3.$
- (c) $P(X_0 = 1, X_1 = 2, X_2 = 3, X_3 = 1) = P(X_3 = 1 | X_0 = 1, X_1 = 2, X_2 = 3) P(X_2 = 3 | X_0 = 1, X_1 = 2) P(X_1 = 2 | X_0 = 1) P(X_0 = 1)$ (by conditioning)
 $= P(X_3 = 1 | X_2 = 3) P(X_2 = 3 | X_1 = 2) P(X_1 = 2 | X_0 = 1) P(X_0 = 1)$ (by the Markov property)
 $= P_{31} P_{23} P_{12} P(X_0 = 1) = (0.8)(0.5)(0.4)(1) = 0.16.$
- (d) We first compute the two-step transition probability matrix. We obtain

$$\mathbf{P}^{(2)} = \begin{bmatrix} 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \\ 0.8 & 0.1 & 0.1 \end{bmatrix} \begin{bmatrix} 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \\ 0.8 & 0.1 & 0.1 \end{bmatrix} = \begin{bmatrix} 0.41 & 0.27 & 0.32 \\ 0.52 & 0.22 & 0.26 \\ 0.34 & 0.36 & 0.30 \end{bmatrix}.$$

Now we write

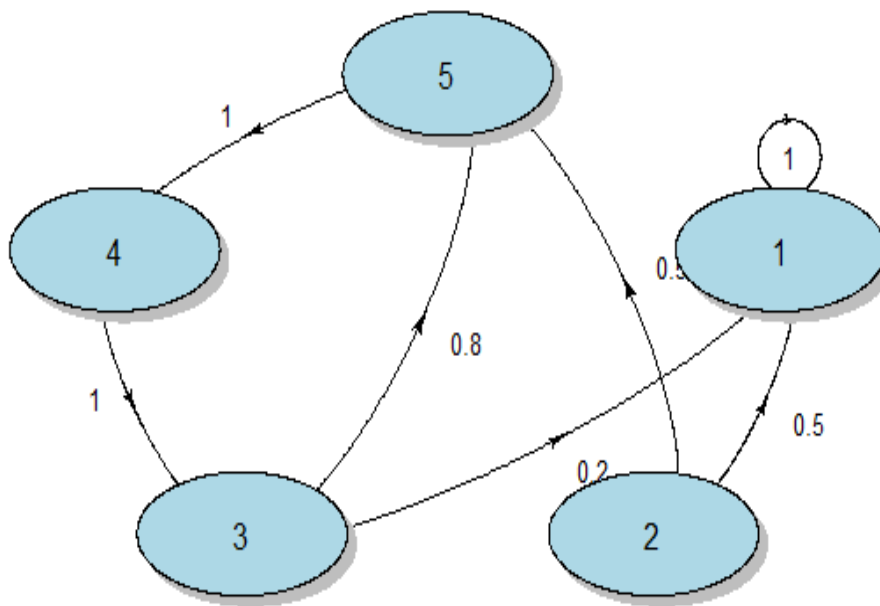
$$\begin{aligned} P(X_0 = 1, X_1 = 2, X_3 = 3, X_5 = 1) &= P(X_5 = 1 | X_0 = 1, X_1 = 2, X_3 = 3) P(X_3 = 3 | X_0 = 1, X_1 = 2) P(X_1 = 2 | X_0 = 1) P(X_0 = 1) \text{ (by conditioning)} \\ &= P(X_5 = 1 | X_3 = 3) P(X_3 = 3 | X_1 = 2) P(X_1 = 2 | X_0 = 1) P(X_0 = 1) \text{ (by the Markov property)} \\ &= P_{31}^{(2)} P_{23}^{(2)} P_{12} P(X_0 = 1) = (0.34)(0.26)(0.4)(1) = 0.03536. \end{aligned}$$

EXERCISE 1.2. (a) We plot a diagram of the Markov chain.

```
#specifying transition probability matrix
tm<- matrix(c(1, 0, 0, 0, 0, 0.5, 0, 0, 0, 0.5, 0.2, 0, 0, 0, 0.8,
0, 0, 1, 0, 0, 0, 0, 0, 1, 0), nrow=5, ncol=5, byrow=TRUE)

#transposing transition probability matrix
tm.tr<- t(tm)

#plotting diagram
library(diagram)
plotmat(tm.tr, arr.length=0.25, arr.width=0.1, box.col="light blue",
box.lwd=1, box.prop=0.5, box.size=0.12, box.type="circle", cex.txt=0.8,
lwd=1, self.cex=0.3, self.shiftx=0.01, self.shifty=0.09)
```



State 2 is reflective. The chain leaves that state in one step. Therefore, it forms a separate transient class that has an infinite period.

Finally, states 3, 4, and 5 communicate and thus belong to the same class. The chain can return to either state in this class in 3, 6, 9, etc. steps, thus the period is equal to 3. Since there is a positive probability to leave this class, it is transient.

The R output supports these findings.

```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm,states=c("1", "2", "3", "4", "5"))

#computing Markov chain characteristics
recurrentClasses(mc)

"1"

transientClasses(mc)

"2"

"3" "4" "5"

absorbingStates(mc)

"1"
```

(c) Below we simulate three trajectories of the chain that start at a randomly chosen state.

```

#specifying total number of steps
nsteps<- 25

#specifying seed
set.seed(4955145)

#specifying initial probability
p0<- c(0.2, 0.2, 0.2, 0.2, 0.2)

#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=3)

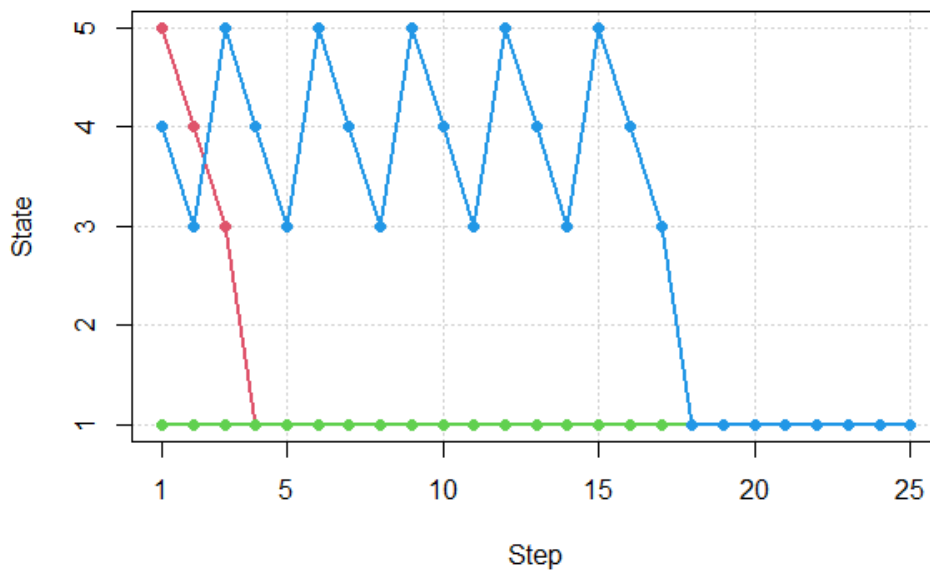
#simulating states
for (i in 1:3) {
  state0<- sample(1:5, 1, prob=p0)
  MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc, t0=state0,
    include.t0=TRUE)
}

#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=2:4, xaxt="n", ylim=c(1,5),
  xlab="Step", ylab="State", panel.first=grid())

axis(side=1, at=c(1,5,10,15,20,25))

points(1:nsteps, MC.states[,1], pch=16, col=2)
points(1:nsteps, MC.states[,2], pch=16, col=3)
points(1:nsteps, MC.states[,3], pch=16, col=4)

```



Since state 1 is an absorbing state, sooner or later, the trajectories transition into this state and don't leave it.

(d) To find the steady-state probabilities, we need to solve the following equations:

$$(\pi_1, \pi_2, \pi_3, \pi_4, \pi_5) = (\pi_1, \pi_2, \pi_3, \pi_4, \pi_5) \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 & 0.5 \\ 0.2 & 0 & 0 & 0 & 0.8 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \text{ with the additional condition}$$

that $\pi_1 + \pi_2 + \pi_3 + \pi_4 + \pi_5 = 1$.

Written out, the system becomes
$$\begin{cases} \pi_1 = \pi_1 + 0.5\pi_2 + 0.2\pi_3 \\ \pi_2 = 0 \\ \pi_3 = \pi_4 = \pi_5 = 0 \\ \pi_1 + \pi_2 + \pi_3 + \pi_4 + \pi_5 = 1 \end{cases}$$
. It has the degenerate solution

$\pi_1 = 1, \pi_2 = \pi_3 = \pi_4 = \pi_5 = 0$. This solution is expected because state 1 is an absorbing state, and so the chain ends up spending 100% of the time there. Having a unique stationary distribution, it is an ergodic Markov chain.

Using R, we obtain:

```
steadyStates(mc)
```

```
1 2 3 4 5
1 0 0 0 0
```

(e) Here we plot the unconditional probabilities at time n against the time.

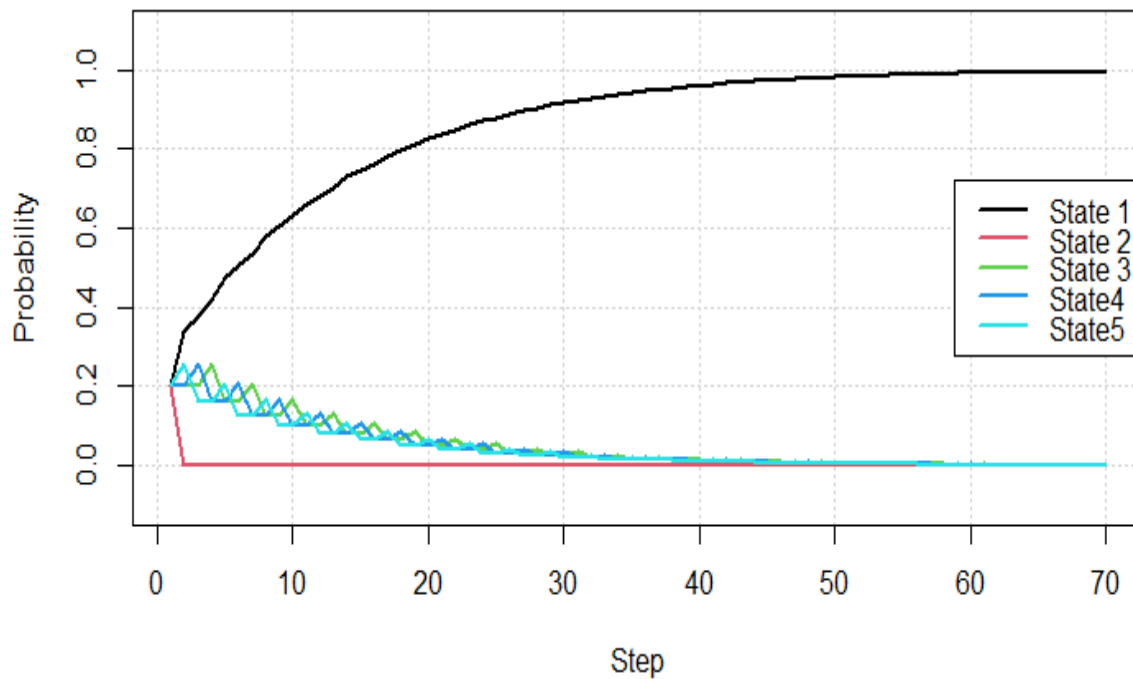
```
#specifying total number of steps
nsteps<- 70
```

```
#specifying matrix containing probabilities
probs<- matrix(NA, nrow=nsteps, ncol=5)
```

```
#computing probabilities
probs[1,] <- p0
for(n in 2:nsteps)
  probs[n,]<- probs[n-1,]%*%tm
```

```
#plotting probabilities vs. step by state
matplot(probs, type="l", lty=1, lwd=2, col=1:5, ylim=c(-0.1, 1.1),
xlab="Step", ylab="Probability", panel.first=grid())
```

```
legend("right", c("State 1", "State 2", "State 3", "State4", "State5"), lty=1,
lwd=2, col=1:5)
```

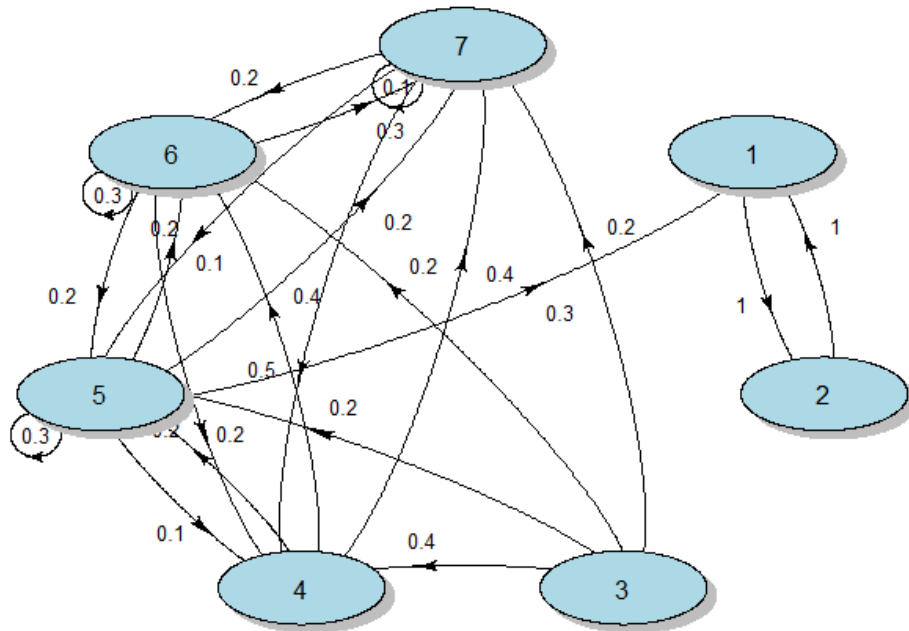


EXERCISE 1.3. (a) We plot a diagram of the Markov chain.

```
#specifying transition probability matrix
tm<- matrix(c(0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0.4,0.2,0.2,0.2,
0,0,0,0,0.2,0.4,0.4,0.3,0,0,0.1,0.3,0.1,0.2,0,0,0,0.2,0.2,0.3,0.3,
0,0,0,0.5,0.2,0.2,0.1),nrow=7, ncol=7, byrow=TRUE)

#transposing transition probability matrix
tm.tr<- t(tm)

#plotting diagram
library(diagram)
plotmat(tm.tr, arr.length=0.3, arr.width=0.1, arr.pos=0.58, box.col="light blue",
box.lwd=1, box.prop=0.5, box.size=0.09, box.type="circle", cex.txt=0.8, lwd=1,
self.cex=0.3, self.shiftx=-0.07, self.shifty=-0.05)
```

(b) States 1 and 2 form a class and it is recurrent. The period is 2. Once the chain transitions into this class, it never leaves it and will bounce between the two states.

State 3 is reflecting. The chain leaves this state in one step. This state forms a class of its own. It is a transient class and its period is infinite.

States 4, 5, 6, and 7 communicate and thus form a class. Its period is one because of the loops. This class is transient because with positive probability the chain can leave this state and transition into the $\{1, 2\}$ class.

From R, we obtain:

```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1", "2", "3", "4", "5",
"6", "7"))

#computing Markov chain characteristics
recurrentClasses(mc)

"1" "2"

transientClasses(mc)

"3"

"4" "5" "6" "7"

absorbingStates(mc)
character(0)

#creating irreducible Markov chain objects
tm.ir<- matrix(c(0,1,1,0),nrow=2, ncol=2, byrow=TRUE)
mc.ir<-new("markovchain", transitionMatrix=tm.ir, states=c("1","2"))
```

```
#finding periods of irreducible Markov chains
period(mc.ir)
```

2

(c) Below we simulate two trajectories of the chain that start at a randomly selected state.

```
#specifying total number of steps
nsteps<- 25

#specifying seed
set.seed(3339964)

#specifying initial probability
p0<- c(1/7, 1/7, 1/7, 1/7, 1/7, 1/7, 1/7)

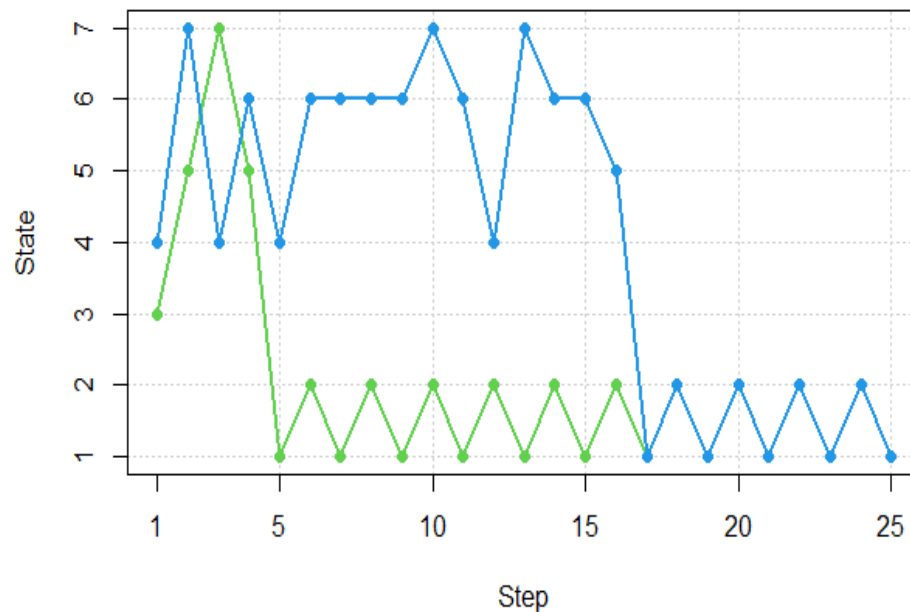
#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=2)

#simulating states
for (i in 1:2){
  state0<- sample(1:7, 1, prob=p0)
  MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc, t0=state0,
    include.t0=TRUE)
}

#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=3:4, ylim=c(1,7), xaxt="n",
  xlab="Step", ylab="State", panel.first=grid())

axis(side=1, at=c(1,5,10,15,20,25))

points(1:nsteps, MC.states[,1], pch=16, col=3)
points(1:nsteps, MC.states[,2], pch=16, col=4)
```



Both simulated trajectories transition to the class $\{1, 2\}$ sooner or later.

(d) Below we calculate the limiting probabilities.

In R:

```
#finding steady-state distribution
round(steadyStates(mc), digits=4)
```

```
  1  2 3 4 5 6 7
0.5 0.5 0 0 0 0 0
```

There is a single limiting distribution which means that the chain is ergodic. States 1 and 2 absorb the chain and then the chain spends 50% of the time in state 1 and the other 50%, in state 2.

(e) Here we plot the unconditional probability vectors p_n against n .

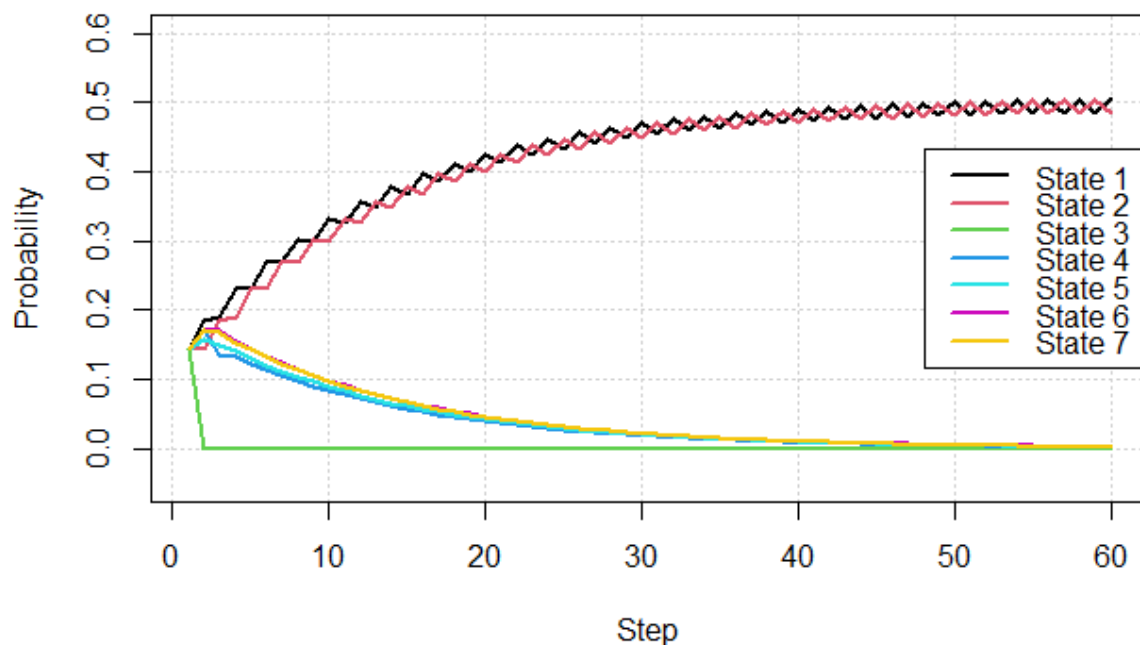
```
#specifying total number of steps
nsteps<- 60
```

```
#specifying matrix containing probabilities
probs<- matrix(NA, nrow=nsteps, ncol=7)
```

```
#computing probabilities
probs[1,] <- p0
for(n in 2:nsteps)
  probs[n,]<- probs[n-1,]%*%tm
```

```
#plotting probabilities vs. step by state
matplot(probs, type="l", lty=1, lwd=2, col=1:7, ylim=c(-0.05, 0.6),
xlab="Step", ylab="Probability", panel.first=grid())
```

```
legend("right", c("State 1","State 2","State 3","State 4","State 5","State 6",
"State 7"), lty=1, lwd=2, col=1:7)state 2","state 3","state 4","state 5","state
6", "state 7"), lty=1, col=1:7)
```



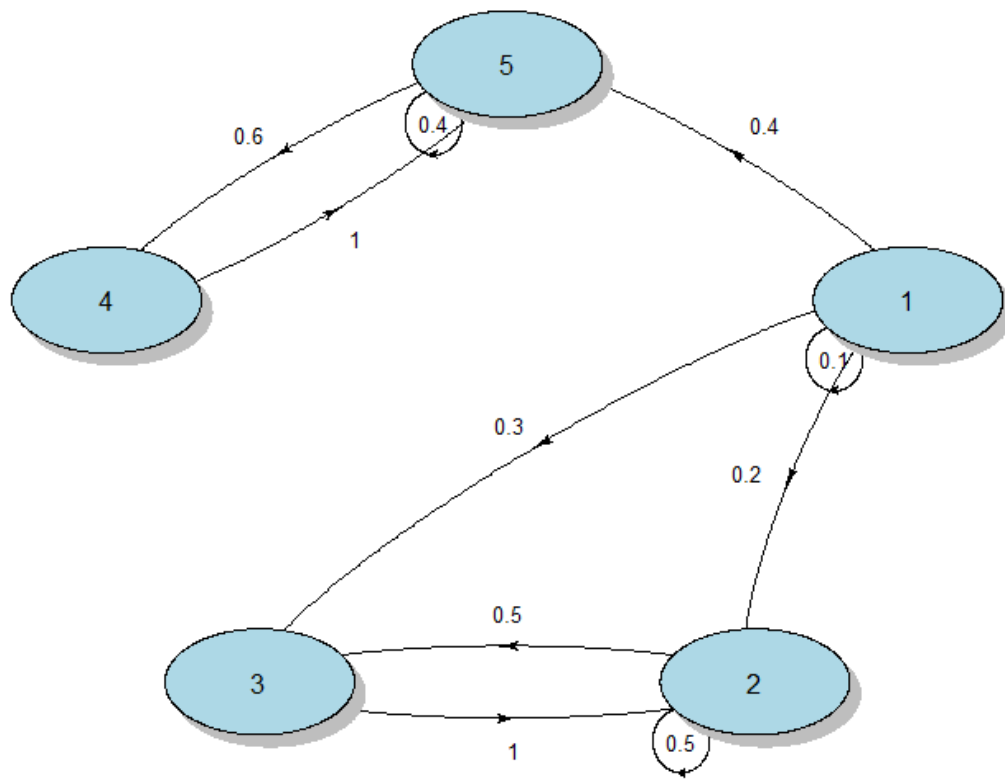
For state 1 and 2 the probabilities converge to 0.5, whereas for all the other states, the probabilities converge to zero. The curves settle around step 50.

EXERCISE 1.4. (a) We plot the diagram of the Markov chain.

```
#specifying the transition probability matrix
tm<- matrix(c(0.1,0.2,0.3,0,0.4,0,0.5,0.5,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0.6,0.4),
nrow=5, ncol=5, byrow=TRUE)

#transposing the transition probability matrix
tm.tr<- t(tm)

#plotting the diagram for the Markov chain
library(diagram)
plotmat(tm.tr, arr.length=0.3, arr.width=0.1, box.col="light blue", box.lwd=1,
box.prop=0.5, box.size=0.09, box.type="circle", cex.txt=0.8, lwd=1, self.cex=0.3,
self.shiftx=-0.07, self.shifty=-0.05)
```



In R:

```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1", "2", "3", "4", "5"))

#computing Markov chain characteristics
recurrentClasses(mc)
```

```
"2" "3"
"4" "5"
```

```
transientClasses(mc)
```

```
"1"
```

```
absorbingStates(mc)
```

```
character(0)
```

```
#creating irreducible Markov chain objects
tm.ir1<- matrix(c(0,1,0.6,0.4),nrow=2, ncol=2, byrow=TRUE)
mc.ir1<-new("markovchain", transitionMatrix=tm.ir, states=c("4","5"))
```

```
#finding periods of irreducible Markov chains
period(mc.ir1)
```

```
1
```

```
#creating irreducible Markov chain objects
tm.ir2<- matrix(c(0.5, 0.5, 1, 0),nrow=2, ncol=2, byrow=TRUE)
mc.ir2<-new("markovchain", transitionMatrix=tm.ir, states=c("2","3"))
```

```
#finding periods of irreducible Markov chains
period(mc.ir2)
```

```
1
```

(c) We simulate two trajectories of the Markov chain.

```
#specifying total number of steps
nsteps<- 25
```

```
#specifying seed
set.seed(202870)
```

```
#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=2)
```

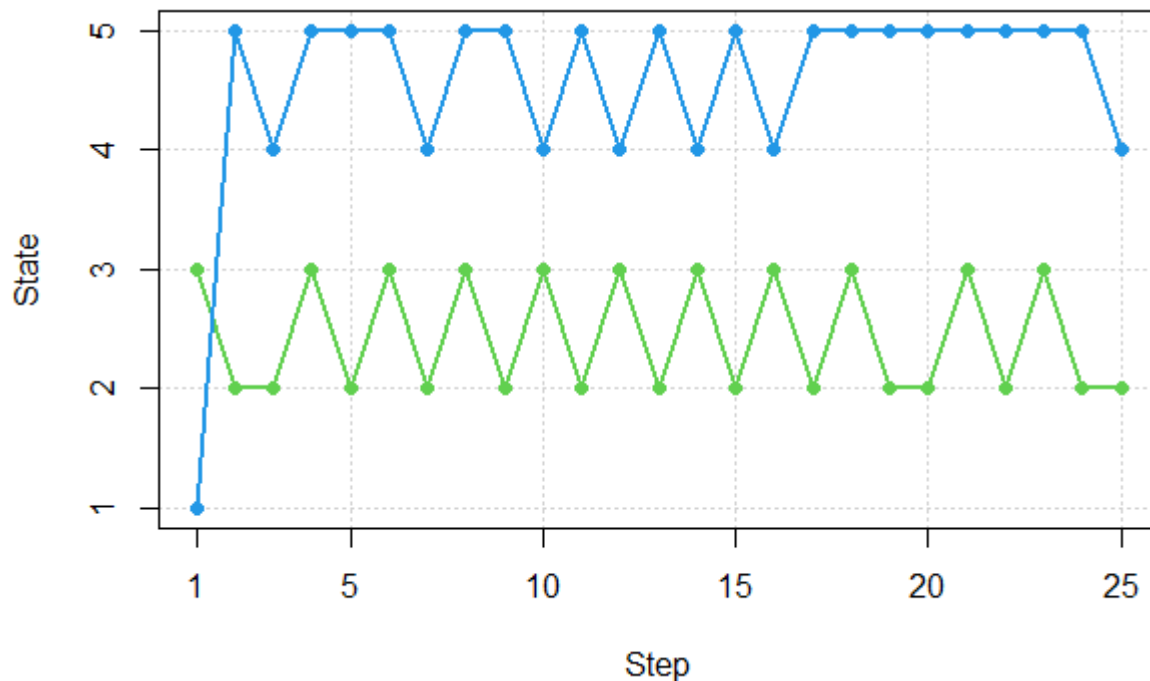
```
#simulating states
for (i in 1:2){
  state0<- sample(1:5, 1, prob=c(1/5, 1/5, 1/5, 1/5, 1/5))
  MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc, t0=state0,
    include.t0=TRUE)
}
```

```
#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=3:4, ylim=c(1,5), xaxt="n",
  xlab="Step", ylab="State", panel.first=grid())
```

```
axis(side=1, at=c(1,5,10,15,20,25))
```

```
points(1:nsteps, MC.states[,1], pch=16, col=3)
points(1:nsteps, MC.states[,2], pch=16, col=4)
```

The trajectories enter either class {2, 3} or {4, 5} and keep bouncing between the two states within each class, possibly remaining for a little bit in state 2 or state 5 because of the loops.



(d) In R, we compute the invariant probability measures.

```
round(steadyStates(mc), digits=4)
```

	1	2	3	4	5
0	0.0000	0.0000	0.375	0.625	
0	0.6667	0.3333	0.000	0.000	

There are two invariant probability measures: $(0, 0, 0, 0.375, 0.625)$ and $(0, 0.6667, 0.3333, 0, 0)$. The chain will settle for one of these distributions, depending on what recurrent class it happens to enter $\{2,3\}$ or $\{4,5\}$. Neither of these two invariant measures is considered to be the stationary distribution because the chain is non-ergodic and the limiting distribution would depend on the initial state of the chain.

(e) We plot the graphs of unconditional probabilities against time, assuming successively that the chain starts in states 1, 2, 3, 4, and 5.

We run the following R code five times, each time changing the initial state.

```
#specifying total number of steps
nsteps<- 20

#specifying matrix containing probabilities
probs<- matrix(NA, nrow=nsteps, ncol=5)

#computing probabilities (initial state 1)
probs[1,] <- c(1,0,0,0,0)
#(state 2) c(0,1,0,0,0) (state 3) c(0,0,1,0,0) (state 4) c(0,0,0,1,0)
#(state 5) c(0,0,0,0,1)
for(n in 2:nsteps)
  probs[n,]<- probs[n-1,]%*%tm

#plotting probabilities vs. step by state
matplot(probs, main="Initial State 1", type="l", lty=1, lwd=2, col=1:5,
```

```
ylim=c(-0.05, 1.1), xlab="Step", ylab="Probability", panel.first=grid())

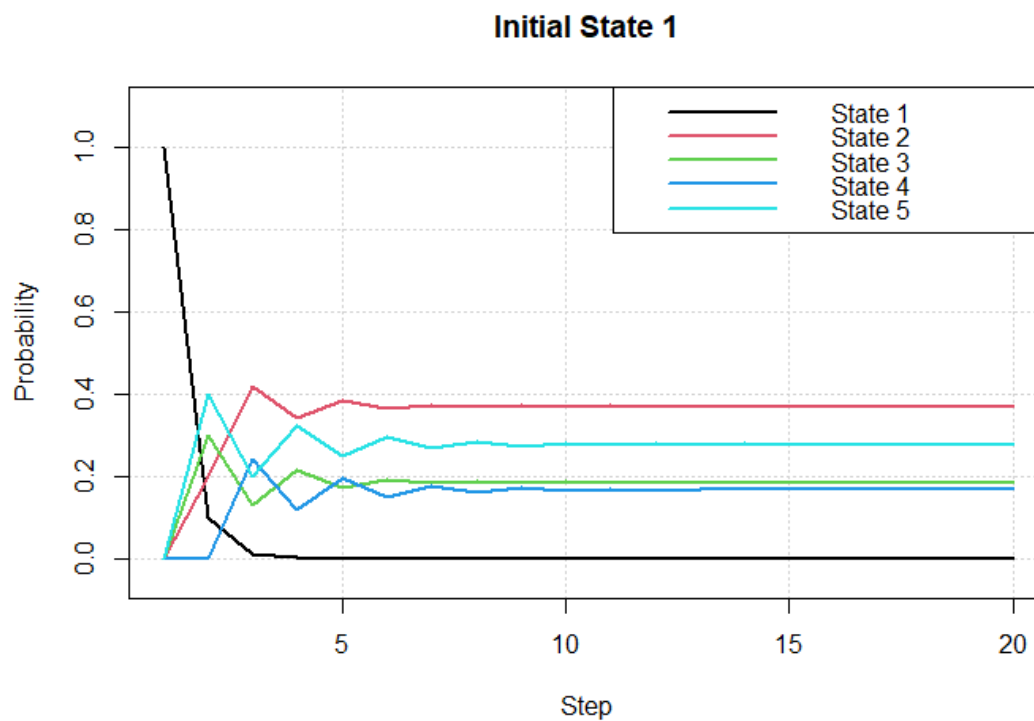
legend("topright", c("State 1", "State 2", "State 3", "State 4", "State 5"),
lty=1, lwd=2, col=1:5)
```

We obtain the following five graphs.

For the initial state 1:

```
> probs

      [,1]      [,2]      [,3]      [,4]      [,5]
[20,] 1e-19 0.3703699 0.1851856 0.1666536 0.2777908
```



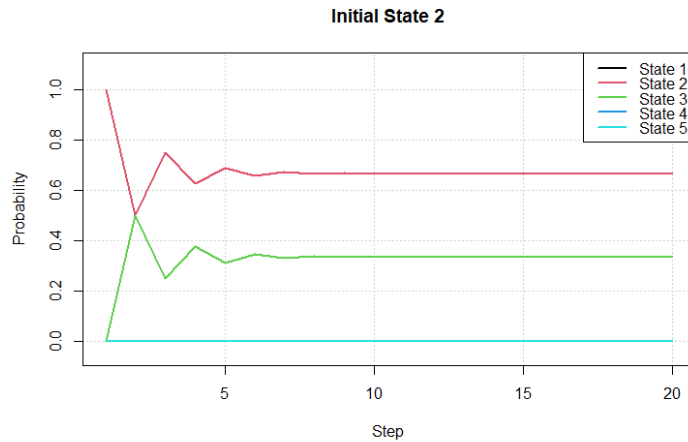
$$\left(\frac{5}{9}\right)(0, 0.6667, 0.3333, 0, 0) + \left(\frac{4}{5}\right)(0, 0, 0, 0.375, 0.625) = (0, 0.3704, 0.1852, 0.1667, 0.2778).$$

For the initial state 2:

```
> probs
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[20,]	0	0.6666660	0.3333340	0	0

If the chain starts in state 2, it will remain within the class $\{2, 3\}$, and the respective probabilities will converge to the invariant vector $(0, 0.6667, 0.3333, 0, 0)$.

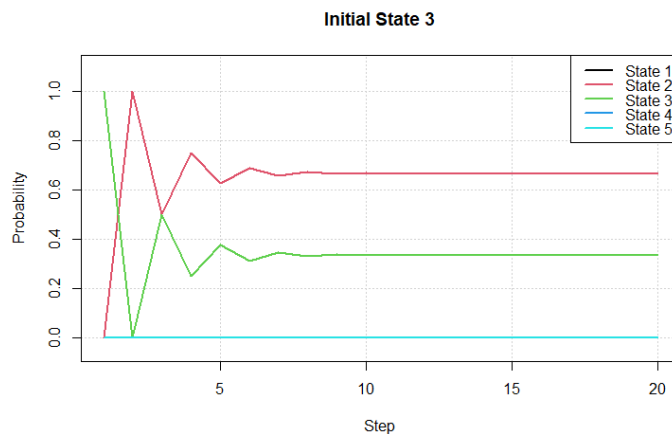


For the initial state 3:

```
> probs
```

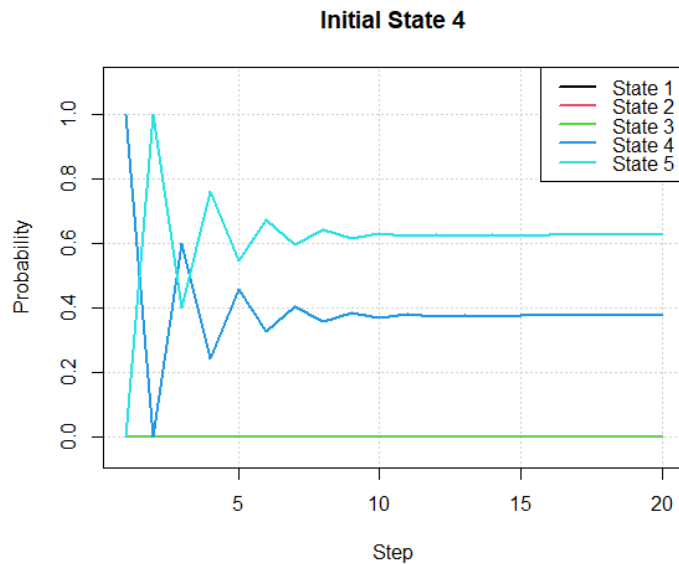
	[,1]	[,2]	[,3]	[,4]	[,5]
[20,]	0	0.6666679	0.3333321	0	0

If the chain starts in state 3, it first goes to state 2 with probability one, and then will transition within the class $\{2, 3\}$, and the respective probabilities will converge to the invariant vector $(0, 0.6667, 0.3333, 0, 0)$.



```
> probs
```

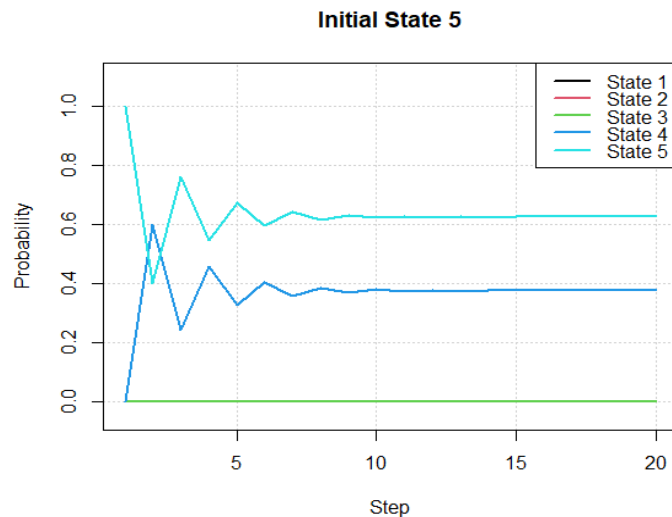
	[,1]	[,2]	[,3]	[,4]	[,5]
[20,]	0	0	0	0.3749619	0.6250381



If the chain starts in state 4, it first transitions into state 5 with probability one, and then will stay within the class $\{4, 5\}$, and the respective probabilities will converge to the invariant vector $(0, 0, 0, 0.375, 0.625)$.

For the initial state 5:

```
> probs
      [,1]      [,2]      [,3]      [,4]      [,5]
[20,]    0         0         0 0.3750229 0.6249771
```



If the chain starts in state 5, it transitions within the class $\{4, 5\}$, and the respective probabilities will converge to the invariant vector $(0, 0, 0, 0.375, 0.625)$.

EXERCISE 1.5. (a) In a box, there are two red (R), four blue (B), and eight green (G) balls. One ball is drawn at a time without replacement and its color is noted. The stochastic process $\{X_n, n = 1, 2, \dots\}$ with the state space $S = \{R, B, G\}$ doesn't satisfy the Markovian property. It can be proved, for example, as $P(X_3 = G | X_1 = R, X_2 = B) = \frac{8}{12}$, whereas $P(X_3 = G | X_1 = G, X_2 = B) = \frac{7}{12}$, thus, the color of the ball drawn at the third step depends on the colors of all previously drawn balls, not just the one drawn at step two.

(b) If the drawing is done with replacement, the process is a Markov chain. Since the balls are put back into the box, the colors of drawn balls are independent of each other. Let C stand for any of the three colors: red, blue, or green. Then we can write

$$P(X_3 = C | X_1 = C, X_2 = C) = P(X_3 = C) = P(X_3 = C | X_2 = C),$$

and thus, the Markov property always holds. Note that a sequence of independent trials is a special case of a Markov chain.

EXERCISE 1.6. Let O denote any outcome of a coin flip. The flips are considered independent, therefore, we obtain

$$P(X_3 = O | X_1 = O, X_2 = O) = P(X_3 = O) = P(X_3 = O | X_2 = O),$$

that is, the Markovian property always holds. The coin is fair, hence, the transition probability matrix is

$$\begin{array}{c} H \\ T \end{array} \begin{array}{cc} H & T \\ \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \end{array}. \text{ To derive the limiting probabilities, we solve } (\pi_H, \pi_T) = (\pi_H, \pi_T) \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$$

where $\pi_H + \pi_T = 1$. The solution is $\pi_H = \pi_T = 0.5$. Note that a sequence of independent trials is a special case of a Markov chain.

EXERCISE 1.7. (a) Below we find the transition probability matrix for Chapter 1 of “Moby Dick” by Herman Melville.

```
library(tidyverse)

chapter1 <- read_file("./Loomings.txt")

#cleaning the text
lowercase<- tolower(chapter1)
no.blanks<- gsub(" ", "", lowercase)
no.line.breaks<- gsub("\r\n", "", no.blanks)
#removing all punctuation
clean.string<- gsub("[[:punct:]]", "", no.line.breaks)

#splitting the string into characters
x2<- strsplit(clean.string, "")

#shifting the text by one place
no.last<- substr(clean.string, 1, nchar(clean.string)-1)
first.blank<- str_c(" ", no.last)
x1<- strsplit(first.blank, "")

vowels<-c("a","e","i","o","u")
consonants<- c("b","c","d","f","g","h","j","k","l","m","n","p","q","r","s","t",
"v","w","x","y","z")

for (counter in 1:nchar(x2)){
  v<- ifelse(x2[[counter]] %in% vowels,1,0)
  c<- ifelse(x2[[counter]] %in% consonants,1,0)
```

```

vv<- ifelse(x1[[counter]] %in% vowels & x2[[counter]] %in% vowels,1,0)
vc<- ifelse(x1[[counter]] %in% vowels & x2[[counter]] %in% consonants,1,0)
cv<- ifelse(x1[[counter]] %in% consonants & x2[[counter]] %in% vowels,1,0)
cc<- ifelse(x1[[counter]] %in% consonants & x2[[counter]] %in% consonants,1,0)
}

```

```
sum(v)
```

3647

```
sum(c)
```

5871

```
sum(vv)
```

572

```
sum(vc)
```

3075

```
sum(cv)
```

3075

```
sum(cc)
```

2795

We check quickly that these numbers add up properly. Since the first chapter of “Moby Dick” starts and ends with consonants, all vowels are transitioned into and transitioned from. Thus, $\text{sum}(v) = 3647 = \text{sum}(vv) + \text{sum}(vc) = 572 + 3075 = \text{sum}(vv) + \text{sum}(cv)$. Also, all but the last consonant are transitioned from, therefore, $\text{sum}(c) - 1 = 5870 = \text{sum}(cv) + \text{sum}(cc) = 3075 + 2795$, and all but the first consonant are transitioned to, so $\text{sum}(c) - 1 = 5870 = \text{sum}(vc) + \text{sum}(cc) = 3075 + 2795$. The transition probability matrix is

$$\begin{array}{cc}
 & \begin{array}{c} v \\ c \end{array} & \begin{array}{c} v \\ c \end{array} \\
 \begin{array}{c} v \\ c \end{array} & \begin{bmatrix} \frac{572}{3647} = 0.15684 & \frac{3075}{3647} = 0.84316 \\ \frac{3075}{5870} = 0.52385 & \frac{2795}{5870} = 0.47615 \end{bmatrix}
 \end{array}$$

The code below computes the limiting probabilities and the proportions of vowels and consonants in the text.

```

#specifying the transition probability matrix
tm<- matrix(c(sum(vv)/sum(v), sum(vc)/sum(v), sum(cv)/(sum(c)-1),
sum(cc)/(sum(c)-1)), nrow=2, ncol=2, byrow=TRUE)

#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("v","c"))

#computing limiting probabilities
steadyStates(mc)

```

0.383209^v 0.616791^c

```
#computing proportions of vowels and consonants
print(prop.vowels<- sum(v)/(sum(v)+sum(c)))
```

0.3831687

```
print(prop.cons<- sum(c)/(sum(v)+sum(c)))
```

0.6168313

From the output, the limiting probabilities are equal to the actual proportions of vowels and consonants.

(b) Now, we run the same code as in part (a), but with the text of Chapter 2. The code and output are

```
library(tidyverse)
chapter2 <- read_file("./The Carpet-Bag.txt")

#cleaning the text
lowercase<- tolower(chapter2)
no.blanks<- gsub(" ","",lowercase)
no.line.breaks<- gsub("\r\n", "", no.blanks)
#removing all punctuation
clean.string<- gsub("[[:punct:]]","",no.line.breaks)

#splitting the string into characters
x2<- strsplit(clean.string, "")

#shifting the text by one place
no.last<- substr(clean.string, 1, nchar(clean.string)-1)
first.blank<- str_c(" ", no.last)
x1<- strsplit(first.blank,"")

vowels<-c("a","e","i","o","u")
consonants<- c("b","c","d","f","g","h","j","k","l","m","n","p","q","r","s","t",
"v","w","x","y","z")

for (counter in 1:nchar(x2)){
  v<- ifelse(x2[[counter]] %in% vowels,1,0)
  c<- ifelse(x2[[counter]] %in% consonants,1,0)
  vv<- ifelse(x1[[counter]] %in% vowels & x2[[counter]] %in% vowels,1,0)
  vc<- ifelse(x1[[counter]] %in% vowels & x2[[counter]] %in% consonants,1,0)
  cv<- ifelse(x1[[counter]] %in% consonants & x2[[counter]] %in% vowels,1,0)
  cc<- ifelse(x1[[counter]] %in% consonants & x2[[counter]] %in% consonants,1,0)
}

sum(v)

2319

sum(c)

3899

sum(vv)
```

340

$\text{sum}(vc)$

1978

$\text{sum}(cv)$

1978

$\text{sum}(cc)$

1921

Chapter 2 starts and ends with vowels, all consonants are transitions into and transitioned from. Therefore, we must have $\text{sum}(c) = 3899 = \text{sum}(vc) + \text{sum}(cc) = 1978 + 1921 = \text{sum}(cv) + \text{sum}(cc)$. Also, all but the last vowel are transitioned from, so $\text{sum}(v) - 1 = 2318 = \text{sum}(vv) + \text{sum}(vc) = 340 + 1978$, and all but the first vowel are transitioned into, so $\text{sum}(v) - 1 = 2318 = \text{sum}(vv) + \text{sum}(cv) = 340 + 1978$. The transition probability matrix is

$$\begin{array}{cc} & \begin{array}{c} v \\ c \end{array} & \begin{array}{c} v \\ c \end{array} \\ \begin{array}{c} v \\ c \end{array} & \left[\begin{array}{cc} \frac{340}{2318} = 0.14668 & \frac{1978}{2318} = 0.85332 \\ \frac{1978}{3899} = 0.50731 & \frac{1921}{3899} = 0.49269 \end{array} \right] \end{array}.$$

These transition probabilities are not exactly equal to the ones in Chapter 1 but are within 1/100th, which is very close.

EXERCISE 1.8. The sentence “The quick brown fox jumped over the lazy dog” repeated 500 times is a deterministic sequence of vowels and consonants and cannot be modeled as a Markov chain. To prove this mathematically, we do the following calculations.

thequickbrownfoxjumpedoverthelazydog|thequick...
ccvcvccccvccccvccvccvccvccvccvccvcc|ccvcvcc...

Let $k = 500$ denote the number of repetitions. There are 0 vvv subsequences, k vvc subsequences, k cvv subsequences, and $10k$ cvc subsequences. Thus, we compute

$$P(X_3 = v | X_2 = v, X_1 = v) = \frac{P(vvv)}{P(vvv) + P(vvc)} = \frac{0}{0+k} = 0, \text{ whereas } P(X_3 = v | X_2 = v) = \frac{P(vvv) + P(cvv)}{P(vvv) + P(cvv) + P(vvc) + P(cvc)} = \frac{0+k}{0+k+k+10k} = \frac{1}{12} \neq 0, \text{ and so, the Markovian property doesn't hold.}$$

EXERCISE 1.9. (a) We show that the genotypes of the direct descendant and the second parent follow a Markov chain with the state space $S = \{(AA, AA), (AA, Aa), (AA, aa), (Aa, AA), (Aa, Aa), (Aa, aa), (aa, AA), (aa, Aa), (aa, aa)\}$ and the transition probability matrix

	(AA, AA)	(AA, Aa)	(AA, aa)	(Aa, AA)	(Aa, Aa)	(Aa, aa)	(aa, AA)	(aa, Aa)	(aa, aa)
(AA, AA)	1/3	1/3	1/3	0	0	0	0	0	0
(AA, Aa)	1/6	1/6	1/6	1/6	1/6	1/6	0	0	0
(AA, aa)	0	0	0	1/3	1/3	1/3	0	0	0
(Aa, AA)	1/6	1/6	1/6	1/6	1/6	1/6	0	0	0
(Aa, Aa)	1/12	1/12	1/12	1/6	1/6	1/6	1/12	1/12	1/12
(Aa, aa)	0	0	0	1/6	1/6	1/6	1/6	1/6	1/6
(aa, AA)	0	0	0	1/3	1/3	1/3	0	0	0
(aa, Aa)	0	0	0	1/6	1/6	1/6	1/6	1/6	1/6
(aa, aa)	0	0	0	0	0	0	1/3	1/3	1/3

- If both parents have the combination AA of genes, their direct descendant will have genes AA with probability one, and will equally likely choose the second parent with genes AA, Aa, or aa. Therefore, the state (AA, AA) transitions into states (AA, AA), (AA, Aa), and (AA, aa) with probabilities equal to 1/3.
- If the parents have genes (AA, Aa), their direct descendant will have genes AA with probability 1/2 or genes Aa with probability 1/2. And will choose the second parent with either of the three types with probability 1/3. Thus, (AA, Aa) transitions into states (AA, AA), (AA, Aa), (AA, aa), (Aa, AA), (Aa, Aa), or (Aa, aa), with probability $\left(\frac{1}{2}\right)\left(\frac{1}{3}\right) = \frac{1}{6}$ each.

(AA, aa), (aa, AA), (aa, Aa), or (aa, aa) with probability $\left(\frac{1}{4}\right)\left(\frac{1}{3}\right) = \frac{1}{12}$ each, and states (Aa, AA), (Aa, Aa), or (Aa, aa) with probability $\left(\frac{1}{2}\right)\left(\frac{1}{3}\right) = \frac{1}{6}$ each.

The other cases are proven similarly.

(b) Below we determine the transient and recurrent classes of the Markov chain.

```
#specifying transition probability matrix
tm<- matrix(c(1/3, 1/3, 1/3, 0, 0, 0, 0, 0, 0, 1/6, 1/6, 1/6, 1/6, 1/6, 1/6, 0, 0,
0, 0, 0, 1/3, 1/3, 1/3, 0, 0, 0, 1/6, 1/6, 1/6, 1/6, 1/6, 1/6, 0, 0, 0,
1/12, 1/12, 1/12, 1/6, 1/6, 1/6, 1/12, 1/12, 1/12, 0, 0, 0, 1/6, 1/6, 1/6, 1/6,
1/6, 1/6, 0, 0, 0, 1/3, 1/3, 1/3, 0, 0, 0, 0, 0, 1/6, 1/6, 1/6, 1/6, 1/6, 1/6,
0, 0, 0, 0, 0, 0, 1/3, 1/3, 1/3), nrow=9, ncol=9, byrow=TRUE)
```

```
#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("AAAA", "AAAa", "AAaa",
"AaAA", "AaAa", "Aaaa", "aaAA", "aaAa", "aaaa"))
```

```
#computing Markov chain characteristics
transientStates(mc)
```

```
character(0)
```

```
recurrentClasses(mc)
```

```
"AAAA" "AAAa" "AAaa" "AaAA" "AaAa" "Aaaa" "aaAA" "aaAa" "aaaa"
```

All states are recurrent and none are transient.

(c) Next, we find the stationary distribution.

```
#finding stationary distribution
steadyStates(mc)
```

```
      AAAA      AAAa      AAaa      AaAA
0.08333333 0.08333333 0.08333333 0.16666667
      AaAa      Aaaa      aaAA      aaAa
0.16666667 0.16666667 0.08333333 0.08333333
      aaaa
0.08333333
```

In a long run, there will be about 8.3% of each gene type (AA, AA) , (AA, Aa) , (AA, aa) , (aa, AA) , (aa, Aa) , and (aa, aa) , and about 16.7% of each gene type (Aa, AA) , (Aa, Aa) , and (Aa, aa) .

(d) Here we find the initial state that achieves the stationary distribution in the smallest number of generations. To answer this question, we run the following R code nine times with the initial state ranging from $(1, 0, 0, \dots, 0)$ to $(0, 0, \dots, 0, 1)$. We look for the smallest generation number for which the steady-state distribution has probabilities that round to 0.0833 and 0.1667.

```
#specifying total number of steps
nsteps<- 20

#specifying matrix containing probabilities
probs<- matrix(NA, nrow=nsteps, ncol=9)

#computing probabilities
probs[1,] <- c(1, 0, 0, 0, 0, 0, 0, 0, 0)

for(n in 2:nsteps) {
  print(n)
  print (probs[n,]<- round(probs[n-1,]%*%tm, 6))
}
```

```
15
[1,] 0.083353 0.083353 0.083353 0.166665 0.166665 0.166665 0.083312 0.083312 0.083312
```

```
16
[1,] 0.083343 0.083343 0.083343 0.166665 0.166665 0.166665 0.083322 0.083322 0.083322
```

The initial state (AA, AA) achieves the steady-state probabilities in the 16th generation. We repeat this code for the other 8 initial states and get the following result:

Initial State	(AA, AA)	(AA, Aa)	(AA, aa)	(Aa, AA)	(Aa, Aa)	(Aa, aa)	(aa, AA)	(aa, Aa)	(aa, aa)
Smallest Generation To Achieve the Steady-State Distribution	16	15	3	15	2	15	3	15	16

The conclusion is that the most genetically diversified type (Aa, Aa) in the first generation results in the limiting distribution already in the second generation, whereas the least diversified genetic types (AA, AA) and (aa, aa) need to wait until the 16th generation to see convergence.

EXERCISE 1.10. The code below selects the weather conditions for Detroit, and computes the empirical conditional probability of clouds tomorrow, given clouds today and yesterday, and the conditional probability of clouds tomorrow, given clouds today.

```
weather.data<- read.csv("./weather_description.csv", header=TRUE, sep=",")
DET<- weather.data$Detroit
```

```
table(DET)
```

```
DET
      broken clouds      drizzle      few clouds
      3975              96      1775
      fog              freezing rain      haze
      585                3      768
heavy intensity drizzle heavy intensity rain heavy shower snow
      8              437      107
      heavy snow light intensity drizzle light intensity drizzle rain
      249              396      2
light intensity shower rain light rain light shower sleet
      193              3873      1
      light shower snow light snow mist
      172              1383      3414
      moderate rain overcast clouds proximity shower rain
      1450              6470      208
proximity thunderstorm scattered clouds shower rain
      104              3940      7
      shower snow sky is clear smoke
      4              15249      1
      snow squalls thunderstorm
      222              1      66
thunderstorm with heavy rain thunderstorm with light rain thunderstorm with rain
      14              41      34
      very heavy rain
      4
```

```
X3<- ifelse(DET=="sky is clear", "no clouds", ifelse(DET %in% c("broken clouds",
"few clouds", "overcast clouds", "scattered clouds", "smoke"), "clouds",
ifelse(DET %in% c("heavy shower snow", "heavy snow", "light shower snow", "light
snow", "shower snow", "snow"), "snow", "rain")))
```

```
table(X3)
```

```
X3
clouds no clouds skyclear snow
16161 11705 15249 2137
```

```
library(Hmisc)
```

```
X2<- Lag(X3,shift=1)
```

```
X1<- Lag(X3,shift=2)
```

```
library(Hmisc)
```

```
X2<- Lag(X3,shift=1)
```

```
X1<- Lag(X3,shift=2)
```

```
#computing P(X3=c|X2=c,X1=c)
```

```
ccc<- ifelse(X1=="clouds" & X2=="clouds" & X3=="clouds",1,0)
```

```
ccn<- ifelse(X1=="clouds" & X2=="clouds" & X3=="no clouds",1,0)
```

```
ccr<- ifelse(X1=="clouds" & X2=="clouds" & X3=="rain",1,0)
```

```
ccs<- ifelse(X1=="clouds" & X2=="clouds" & X3=="snow",1,0)
```

```
sum(ccc)/sum(ccc+ccn+ccr+ccs)
```

0.8143572

```
#computing P(X3=c|X2=c)
```

```
ncc<- ifelse(X1=="no clouds" & X2=="clouds" & X3=="clouds",1,0)
```

```
rcc<- ifelse(X1=="rain" & X2=="clouds" & X3=="clouds",1,0)
```



```

scc<- ifelse(X1=="snow" & X2=="clouds" & X3=="clouds",1,0)
ncn<- ifelse(X1=="no clouds" & X2=="clouds" & X3=="no clouds",1,0)
ncr<- ifelse(X1=="no clouds" & X2=="clouds" & X3=="rain",1,0)
ncs<- ifelse(X1=="no clouds" & X2=="clouds" & X3=="snow",1,0)
rcn<- ifelse(X1=="rain" & X2=="clouds" & X3=="no clouds",1,0)
rcr<- ifelse(X1=="rain" & X2=="clouds" & X3=="rain",1,0)
rcs<- ifelse(X1=="rain" & X2=="clouds" & X3=="snow",1,0)
scn<- ifelse(X1=="snow" & X2=="clouds" & X3=="no clouds",1,0)
scr<- ifelse(X1=="snow" & X2=="clouds" & X3=="rain",1,0)
scs<- ifelse(X1=="snow" & X2=="clouds" & X3=="snow",1,0)

sum(ccc+ncc+rcc+scc)/sum(ccc+ccn+ccr+ccs+ncc+ncn+ncr+ncs+rcc+rcn
+rcr+rcs+scc+scn+scr+scs)

```

0.7662892

The state space of this process is $S = \{c = \text{"clouds"}, n = \text{"no clouds"}, r = \text{"rain"}, s = \text{"snow"}\}$. From the above output,

$$\begin{aligned}
 P(X_3 = c \mid X_1 = c, X_2 = c) &= \frac{P(X_1 = c, X_2 = c, X_3 = c)}{P(X_1 = c, X_2 = c)} \\
 &= \frac{P(ccc)}{P(ccc) + P(ccn) + P(ccr) + P(ccs)} = 0.8143572,
 \end{aligned}$$

whereas

$$P(X_3 = c \mid X_2 = c) = \frac{P(ccc) + P(ncc) + P(rcc) + P(scc)}{P(ccc) + P(ccn) + P(ccr) + \dots + P(scs)} = 0.7662892.$$

Since the two quantities are not the same, the process is not a Markov chain.

EXERCISE 1.11. Let s denote the air quality status (good/unhealthy/hazardous). We are given that $P(X_{n+1} = s \mid X_n = s, X_{n-1} = s, \dots, X_1 = s) = P(X_{n+1} = s \mid X_n = s, X_{n-1} = s)$. We consider states as air quality statuses in two consecutive days: (X_1, X_2) , (X_2, X_3) , etc. We show that the Markov property holds:

$$\begin{aligned}
 &P((X_n, X_{n+1}) = (s, s) \mid (X_{n-1}, X_n) = (s, s), \dots, (X_1, X_2) = (s, s)) \\
 &= P(X_{n+1} = s, X_n = s \mid X_n = s, X_{n-1} = s, \dots, X_1 = s) \\
 &= \frac{P(X_{n+1} = s, X_n = s, X_{n-1} = s, \dots, X_1 = s)}{P(X_n = s, X_{n-1} = s, \dots, X_1 = s)} \\
 &= P(X_{n+1} = s \mid X_n = s, X_{n-1} = s, \dots, X_1 = s) \\
 &= P(X_{n+1} = s \mid X_n = s, X_{n-1} = s) \text{ (by the assumption of the problem)} \\
 &= \frac{P(X_{n+1} = s, X_n = s, X_{n-1} = s)}{P(X_n = s, X_{n-1} = s)} \\
 &= P((X_n, X_{n+1}) = (s, s) \mid (X_{n-1}, X_n) = (s, s)).
 \end{aligned}$$

EXERCISE 1.12. (a) Let state 1 give income \$200, state 2 give income \$0, state 3 give income -\$75, state 4 give income \$105, and state 5 give income -\$130. Since the rolls of the die are independent, the next state will depend only on the present state. We use the fact that the die is fair and that the states are traversed circularly, and write the transition probability matrix:

	1	2	3	4	5
1	1/6	1/3	1/6	1/6	1/6
2	1/6	1/6	1/3	1/6	1/6
3	1/6	1/6	1/6	1/3	1/6
4	1/6	1/6	1/6	1/6	1/3
5	1/3	1/6	1/6	1/6	1/6

(b) We compute the steady-state probability of each square and find the long-run winning of the player.

```
#specifying transition probability matrix
tm<- matrix(c(1/6, 1/3, 1/6, 1/6, 1/6,
              1/6, 1/6, 1/3, 1/6, 1/6,
              1/6, 1/6, 1/6, 1/3, 1/6,
              1/6, 1/6, 1/6, 1/6, 1/3,
              1/3, 1/6, 1/6, 1/6, 1/6),
            nrow=5, ncol=5, byrow=TRUE)

#creating Markov chain object
library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1", "2", "3", "4", "5"))

steadyStates(mc)

      1      2      3      4      5
0.2 0.2 0.2 0.2 0.2
```

In the long-run, the chain will be uniformly distributed between the five states, and thus the expected winning of the player will be $E(\text{winning}) = (\$200)(0.2) + (\$0)(0.2) + (-\$75)(0.2) + (\$105)(0.2) + (-\$130)(0.2) = \20 .

EXERCISE 1.13. (a) Assuming that the traffic starts with the light state at 1PM, we find the distribution of the states at 6PM. Traffic conditions change every 20 minutes, therefore between 1PM and 4PM there will be 9 transitions between the states (light/heavy/jammed), and between 4PM and 6PM there will be 6 transitions. We run the following R code to find the distribution of states at 6PM:

```
#specifying the transition probability matrices
tm1<- matrix(c(0.4, 0.4, 0.2, 0.3, 0.5, 0.2, 0, 0.5, 0.5), nrow=3, ncol=3,
              byrow=TRUE)
tm2<- matrix(c(0.1, 0.5, 0.4, 0.1, 0.3, 0.6, 0, 0.1, 0.9), nrow=3, ncol=3,
              byrow=TRUE)

#computing the unconditional distribution at 6pm
library(expm)
state1pm<- c(1, 0, 0)
state4pm<- state1pm%*%(tm1%^9)
print(state6pm<- state4pm%*%(tm2%^6))

      [,1]      [,2]      [,3]
0.01504877 0.1332313 0.85172
```

$P(\text{light traffic}) = 0.01504877$, $P(\text{heavy traffic}) = 0.1332313$, and $P(\text{jammed traffic}) = 0.85172$.

(b) Below we simulate 10,000 trajectories to verify the result of part (a).

```
#creating Markov chain objects
library(markovchain)
mc1<- new("markovchain", transitionMatrix=tm1, states=c("light", "heavy",
"jammed"))

mc2<- new("markovchain", transitionMatrix=tm2, states=c("light", "heavy",
"jammed"))

#simulating states between 1pm and 4pm
MC.states4pm<- matrix(NA, nrow=9, ncol=10000)

for (i in 1:10000)
MC.states4pm[,i]<- rmarkovchain(n=9, object=mc1, t0="light")

#simulating states between 4pm and 6pm
MC.states6pm<- matrix(NA, nrow=6, ncol=10000)

for (i in 1:10000)
MC.states6pm[,i]<- rmarkovchain(n=6, object=mc2, t0=MC.states4pm[9,i])

#concatenating two matrices
MC.states<- rbind(MC.states4pm, MC.states6pm)

#computing frequencies of states at 6pm
table(MC.states[15,])

heavy jammed light
1316 8533 151
```

Thus, the estimates are $\hat{P}(\text{light traffic}) = 0.0151$, $\hat{P}(\text{heavy traffic}) = 0.1316$, and $\hat{P}(\text{jammed traffic}) = 0.8533$.

EXERCISE 1.14. (a) Assuming that a shrub is initially sustainable, we simulate three trajectories of the Markov chain.

```
#specifying transition probability matrix
tm<- matrix(c(0.6, 0.2, 0.1, 0.1, 0.7, 0.2, 0.1, 0, 0.1, 0.3, 0.4, 0.2, 0,
0, 0, 1), nrow=4, ncol=4, byrow=TRUE)

library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("1", "2", "3", "4"))

#specifying total number of steps
nsteps<- 25

#specifying seed
set.seed(912332)

#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=3)
```

```

#simulating states

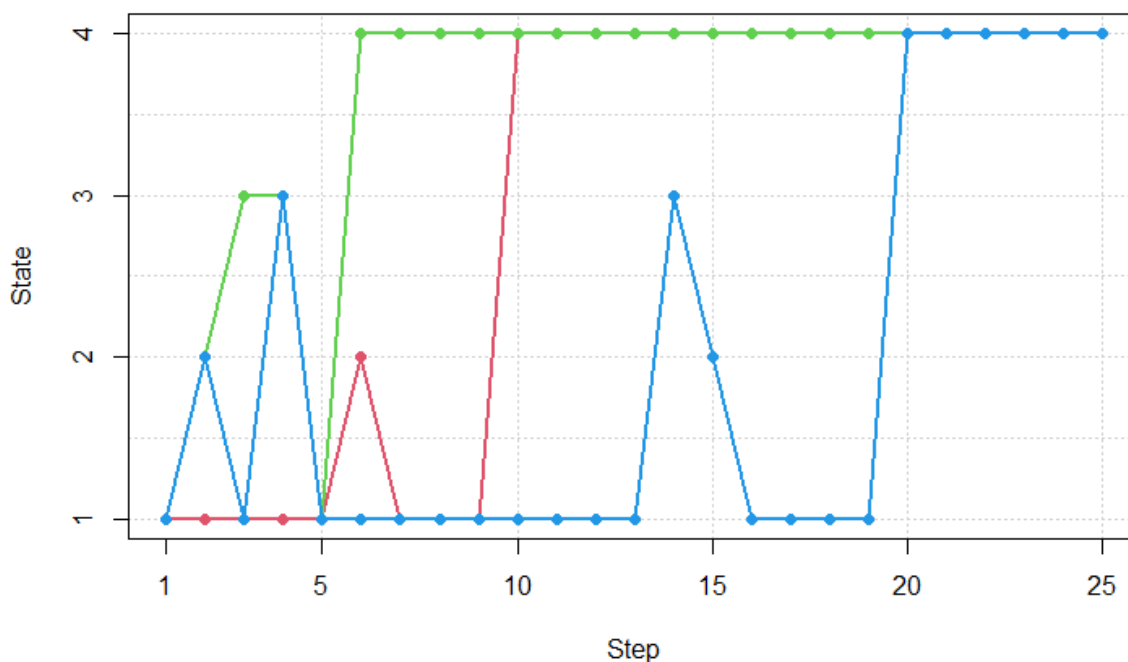
for (i in 1:3){
  state0<- 1
  MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc, t0=state0,
    include.t0=TRUE)
}

#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=2:4, xaxt="n", yaxt="n",
  ylim=c(1,4), xlab="Step", ylab="State", panel.first=grid())

axis(side=1, at=c(1,5,10,15,20,25))
axis(side=2, at=c(1,2,3,4))

points(1:nsteps, MC.states[,1], pch=16, col=2)
points(1:nsteps, MC.states[,2], pch=16, col=3)
points(1:nsteps, MC.states[,3], pch=16, col=4)

```



(b) Here we find the probability that an initially sustainable shrub will eventually become extinct.

```
steadyStates(mc)
```

```

1 2 3 4
0 0 0 1

```

State 4 (“extinct”) is an absorbing state, thus, eventually, a sustainable shrub will become extinct with probability one.

EXERCISE 1.15. (a) The number of music instruments on Tuesday morning can assume values 3, 4, 5, 6, or 7. Therefore, the state space consists of these five states.

Let $N \sim \text{Poisson}(4)$ be the number of instruments bought during the week. Then

$$\begin{aligned}
P(3 \text{ instruments this week} \mid 3 \text{ instruments previous week}) &= P(N = 0) = \exp(-4) = 0.0183 \\
P(4 \text{ instruments this week} \mid 3 \text{ instruments previous week}) &= 0 \\
P(5 \text{ instruments this week} \mid 3 \text{ instruments previous week}) &= 0 \\
P(6 \text{ instruments this week} \mid 3 \text{ instruments previous week}) &= 0 \\
P(7 \text{ instruments this week} \mid 3 \text{ instruments previous week}) &= P(N \geq 1) = 1 - 0.0183 = 0.9817
\end{aligned}$$

$$\begin{aligned}
P(3 \text{ instruments this week} \mid 4 \text{ instruments previous week}) &= P(N = 1) = 4 \exp(-4) = 0.0733 \\
P(4 \text{ instruments this week} \mid 4 \text{ instruments previous week}) &= P(N = 0) = \exp(-4) = 0.0183 \\
P(5 \text{ instruments this week} \mid 4 \text{ instruments previous week}) &= 0 \\
P(6 \text{ instruments this week} \mid 4 \text{ instruments previous week}) &= 0 \\
P(7 \text{ instruments this week} \mid 4 \text{ instruments previous week}) &= P(N \geq 2) = 1 - 0.0733 - 0.0183 \\
&= 0.9084
\end{aligned}$$

$$\begin{aligned}
P(3 \text{ instruments this week} \mid 5 \text{ instruments previous week}) &= P(N = 2) = \frac{4^2}{2!} \exp(-4) = 0.1465 \\
P(4 \text{ instruments this week} \mid 5 \text{ instruments previous week}) &= P(N = 1) = 4 \exp(-4) = 0.0733 \\
P(5 \text{ instruments this week} \mid 5 \text{ instruments previous week}) &= P(N = 0) = \exp(-4) = 0.0183 \\
P(6 \text{ instruments this week} \mid 5 \text{ instruments previous week}) &= 0 \\
P(7 \text{ instruments this week} \mid 5 \text{ instruments previous week}) &= P(N \geq 3) = 1 - 0.1465 - 0.0733 \\
&- 0.0183 = 0.7619
\end{aligned}$$

$$\begin{aligned}
P(3 \text{ instruments this week} \mid 6 \text{ instruments previous week}) &= P(N = 3) = \frac{4^3}{3!} \exp(-4) = 0.1954 \\
P(4 \text{ instruments this week} \mid 6 \text{ instruments previous week}) &= P(N = 2) = \frac{4^2}{2!} \exp(-4) = 0.1465 \\
P(5 \text{ instruments this week} \mid 6 \text{ instruments previous week}) &= P(N = 1) = 4 \exp(-4) = 0.0733 \\
P(6 \text{ instruments this week} \mid 6 \text{ instruments previous week}) &= P(N = 0) = \exp(-4) = 0.0183 \\
P(7 \text{ instruments this week} \mid 6 \text{ instruments previous week}) &= P(N \geq 4) = 1 - 0.1954 - 0.1465 \\
&- 0.0733 - 0.0183 = 0.5665
\end{aligned}$$

$$\begin{aligned}
P(3 \text{ instruments this week} \mid 7 \text{ instruments previous week}) &= P(N = 4) = \frac{4^4}{4!} \exp(-4) = 0.1954 \\
P(4 \text{ instruments this week} \mid 7 \text{ instruments previous week}) &= P(N = 3) = \frac{4^3}{3!} \exp(-4) = 0.1954 \\
P(5 \text{ instruments this week} \mid 7 \text{ instruments previous week}) &= P(N = 2) = \frac{4^2}{2!} \exp(-4) = 0.1465 \\
P(6 \text{ instruments this week} \mid 7 \text{ instruments previous week}) &= P(N = 1) = 4 \exp(-4) = 0.0733 \\
P(7 \text{ instruments this week} \mid 7 \text{ instruments previous week}) &= P(N = 0) + P(N \geq 5) = \\
&1 - 0.1954 - 0.1954 - 0.1465 - 0.0733 = 0.5665 = 0.3894
\end{aligned}$$

The one-step transition probability matrix is

	3	4	5	6	7
3	0.0183	0	0	0	0.9817
4	0.0733	0.0183	0	0	0.9084
5	0.1465	0.0733	0.0183	0	0.7619
6	0.1954	0.1465	0.0733	0.0183	0.5665
7	0.1954	0.1954	0.1465	0.0733	0.3894

(b) The following code generates inventory trajectories, assuming that the initial inventory size is randomly chosen.

```
#specifying transition probability matrix
tm<- matrix(c(0.0183, 0, 0, 0, 0.9817, 0.0733, 0.0183, 0, 0, 0.9084, 0.1465,
0.0733, 0.0183, 0, 0.7619, 0.1954, 0.1465, 0.0733, 0.0183, 0.5665, 0.1954,
0.1954, 0.1465, 0.0733, 0.3894), nrow=5, ncol=5, byrow=TRUE)

library(markovchain)
mc<- new("markovchain", transitionMatrix=tm, states=c("3", "4", "5", "6", "7"))

#specifying total number of steps
nsteps<- 25

#specifying seed
set.seed(8596943)

#specifying matrix containing states
MC.states<- matrix(NA, nrow=nsteps, ncol=2)

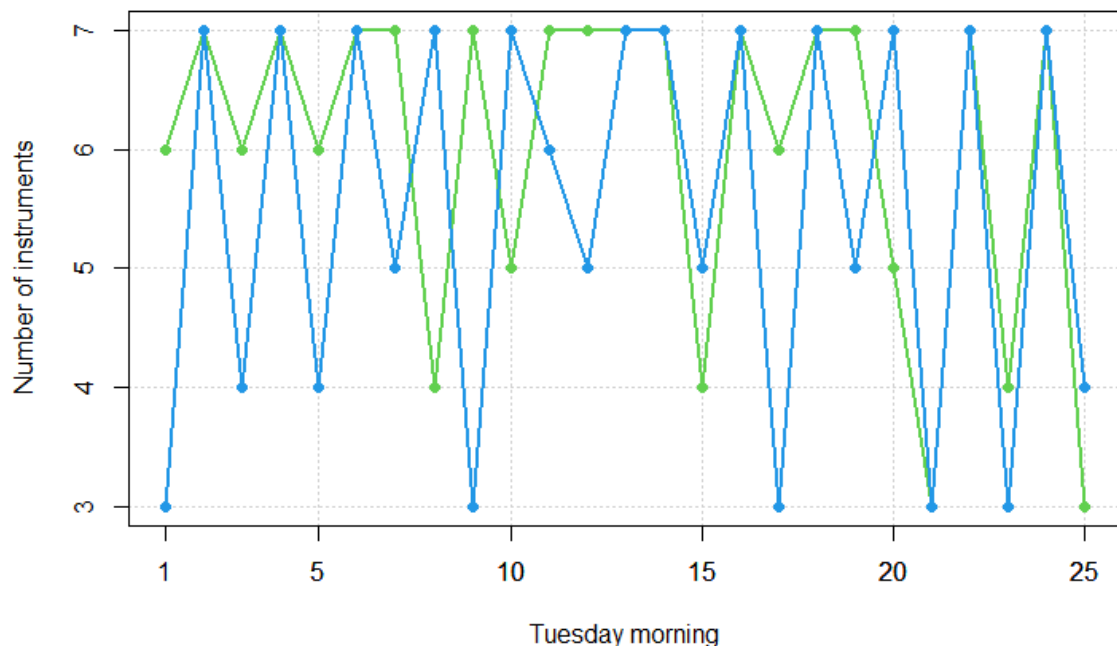
#simulating states

for (i in 1:2) {
  state0<- sample(3:7, 1, prob=c(0.2,0.2,0.2,0.2,0.2))
  MC.states[,i]<- rmarkovchain(n=nsteps-1, object=mc, t0=state0,
include.t0=TRUE)
}

#plotting simulated trajectories
matplot(MC.states, type="l", lty=1, lwd=2, col=3:4, axat="n", ylim=c(3,7),
xlab="Tuesday morning", ylab="Number of instruments", panel.first=grid())

axis(side=1, at=c(1,5,10,15,20,25))

points(1:nsteps, MC.states[,1], pch=16, col=3)
points(1:nsteps, MC.states[,2], pch=16, col=4)
```



(c) Using the definition of conditional probability and Markov property, we can write

$$\begin{aligned}
 P(X_4 = 7, X_3 = 7, X_2 = 7 \mid X_1 = 7) &= \frac{P(X_4 = 7, X_3 = 7, X_2 = 7, X_1 = 7)}{P(X_1 = 7)} \\
 &= \frac{P(X_4 = 7 \mid X_3 = 7, X_2 = 7, X_1 = 7) P(X_3 = 7, X_2 = 7, X_1 = 7)}{P(X_1 = 7)} \\
 &= \frac{P(X_4 = 7 \mid X_3 = 7) P(X_3 = 7 \mid X_2 = 7, X_1 = 7) P(X_2 = 7, X_1 = 7)}{P(X_1 = 7)} \\
 &= \frac{P(X_4 = 7 \mid X_3 = 7) P(X_3 = 7 \mid X_2 = 7) P(X_2 = 7 \mid X_1 = 7) P(X_1 = 7)}{P(X_1 = 7)} \\
 &= P_{77}P_{77}P_{77} = P_{77}^3 = (0.3894)^3 = 0.059.
 \end{aligned}$$

(d) Below we find the steady-state probability distribution for this Markov chain.

steadyStates(mc)

0.1487365³ 0.1300719⁴ 0.09080707⁵ 0.04379828⁶ 0.5865863⁷

Using these values, we compute the expected weekly storage cost:

$$\begin{aligned}
 E(\text{weekly storage cost}) &= \$5 \left((3)(0.1487365) + (4)(0.1300719) + (5)(0.09080707) \right. \\
 &\quad \left. + (6)(0.04379828) + (7)(0.5865863) \right) = (\$5)(5.789426) = \$28.94713.
 \end{aligned}$$

Thus, in the long run, there will be, on average, 5.789426 instruments in the store on Tuesday morning, and the average storage cost will amount to \$28.95.

CHAPTER 2

EXERCISE 2.1. In theory, $E(X_{50}) = ((2)(0.3) - 1)(50) = -20$ and $Var(X_{50}) = (4)(0.3)(1 - 0.3)(50) = 42$.

Next, we run an R code that simulates 10,000 trajectories of length 50 steps and computes the mean and variance of the last values.

```
#specifying parameters
p<- 0.3
n<- 50
ntraj<- 10000

#setting seed number
set.seed(546675)

#defining walk as matrix
walk<- matrix(NA, nrow=n, ncol=ntraj)

#simulating trajectories
for (j in 1:ntraj) {
  walk[1,j]<- 0
  for (k in 2:n) {
    walk[k,j]<- ifelse(runif(1)<p, walk[k-1,j]+1, walk[k-1,j]-1)
  }
}

mean(walk[50,])

-19.5824

var(walk[50,])

42.16583
```

The empirical values are pretty close to the theoretical ones.

EXERCISE 2.2. (a) The R script below simulates 10,000 trajectories and counts how many of them have a value of 0 at the 1,000th step.

```
#setting counter to zero
nzeros<- 0

#specifying seed
set.seed(675572)

#defining walk as matrix
walk<- c()

#simulating trajectories
for (j in 1:10000)
{
  walk[1]<- 0
  for (i in 2:1001)
```



```

walk[i]<- ifelse(runif(1)<0.5, walk[i-1]+1, walk[i-1]-1)
  if (walk[1001]==0) nzeros=nzeros+1
}

print(nzeros)

```

253

(a) The theoretical probability of returning to 0 on the 1,000th step is

$$P(X_{1000} = 0 \mid X_0 = 0) = \binom{1000}{500} \left(\frac{1}{2}\right)^{1000} = 0.025. \text{ This quantity was computed in R:}$$

```
choose(1000, 500)*0.5^1000
```

0.02522502

The estimated probability from part (a) is $\hat{P}(X_{1000} = 0 \mid X_0 = 0) = \frac{253}{10000} = 0.0253$, which is a pretty accurate estimate of the theoretical value.

EXERCISE 2.3. (a) The code below simulates the 10,000 trajectories of one-, two-, and three-dimensional symmetric random walks that start at the origin and continue for at most 1,000 steps. A trajectory that reaches the origin is terminated.

```

#setting counters to zero
n1D<- 0
n2D<- 0
n3D<- 0

#specifying seed
set.seed(300799)

#defining 1D walk as vector
walk1D<- c()
nsteps1D<- c()

#simulating 1D trajectories
for (j in 1:10000)
{
  walk1D[1]<- 0 #setting initial value to zero
  for (i in 2:1001)
  {
    walk1D[i]<- ifelse(runif(1)<0.5, walk1D[i-1]+1, walk1D[i-1]-1)
    if (walk1D[i]==0) {
      n1D=n1D+1
      break
    }
  }
  nsteps1D[j]=i
}

#defining 2D walk as matrix
walk2D<- matrix(NA, nrow=1001, ncol=2)
nsteps2D<- c()

#defining random steps

```

```

rstep2D<- matrix(c(1, 0, -1, 0, 0, 1, 0, -1), nrow=4, ncol=2, byrow=TRUE)

#simulating 2D trajectories
for (j in 1:10000)
{
  walk2D[1,]<- c(0,0) #setting initial value to the origin
  for (i in 2:1001)
  {
    walk2D[i,]<- walk2D[i-1,]+rstep2D[sample(1:4, size=1),]

    if (walk2D[i,1]==0 & walk2D[i,2]==0) {
      n2D=n2D+1
      break
    }
  }
  nsteps2D[j]=i
}

#defining 3D walk as matrix
walk3D<- matrix(NA, nrow=1001, ncol=3)
nsteps3D<- c()

#defining random steps
rstep3D<- matrix(c(1, 0, 0,-1, 0, 0, 0, 1, 0, 0, -1, 0, 0, 0, 1, 0, 0, -1),
nrow=6, ncol=3, byrow=TRUE)

#simulating 3D trajectories
for (j in 1:10000)
{
  walk3D[1,]<- c(0,0,0) #setting initial value to the origin
  for (i in 2:1001)
  {
    walk3D[i,]<- walk3D[i-1,]+rstep3D[sample(1:6, size=1),]

    if (walk3D[i,1]==0 & walk3D[i,2]==0 & walk3D[i,3]==0) {
      n3D=n3D+1
      break
    }
  }
  nsteps3D[j]=i
}

print(n1D)

9756

print(n2D)

6759

print(n3D)

3329

```

Roughly 97.6% of the 1D trajectories returned to 0, about 67.6% of the 2D trajectories returned to (0, 0), and only 33.3% of the 3D trajectories returned to (0, 0, 0).

(a) The average number of steps it took those trajectories to return to the origin is computed as

```
mean(nsteps1D[nsteps1D!=1001])
```

27.47499

```
mean(nsteps2D[nsteps2D!=1001])
```

61.47625

```
mean(nsteps3D[nsteps3D!=1001])
```

27.83689

The 97.6% of the 1D trajectories that returned to the origin, did it in 27.47 steps, on average.
The 67.6% of the 2D trajectories that returned to the origin, did it in 61.48 steps, on average.
The 33.3% of the 3D trajectories that returned to the origin, did it in 27.84 steps, on average.

EXERCISE 2.4. (a) The R script below simulates the trajectories and terminates them if the barrier is hit. Otherwise, trajectories continue for 1,000 steps. The total number of trajectories that hit the barrier is counted. We also record the number of steps (for part (b)) and the y-coordinate (for part (c)).

```
#setting counter to zero
nhits<- 0

#specifying seed
set.seed(50118)

#defining walk as matrix
walk<- matrix(NA, nrow=1001, ncol=2)
nsteps<- c()
ycoord<- c()

#defining random steps
rstep<- matrix(c(1, 0, -1, 0, 0, 1, 0, -1), nrow=4, ncol=2, byrow=TRUE)

#simulating trajectories
for (j in 1:10000)
{
  walk[1,]<- c(0,0) #setting initial value to the origin
  for (i in 2:1001)
  {
    walk[i,]<- walk[i-1,] + rstep[sample(1:4, size=1),]

    if (walk[i,1]==30) {
      nhits=nhits+1
      break
    }
  }
  nsteps[j]<- i
  ycoord[j]<- ifelse(i==1001, 99999, walk[i,2])
}

print(nhits)
```

1764

So, of the 10,000 trajectories, 1,764 hit the vertical barrier. Thus, the estimated probability to hit the barrier is 0.1764.

(b) The average number of steps it takes a trajectory to hit the barrier, provided it did hit the barrier within the 1,000 steps, is estimated as

```
mean(nsteps[nsteps!=1001])
```

623.2053

It took on average 623.2 steps to hit the barrier for the 17.64% of the trajectories that terminated at the barrier.

(a) Estimate the expected value of the y-coordinate at the time when the random walk hits the barrier. What should this value be from the theoretical point of view? Hint: deduce from a symmetry argument.

```
mean(ycoord[ycoord!=99999])
```

0.1066364

The estimated average y-coordinate for 17.64% of the trajectories that hit the barrier was 0.1066. From the theoretical viewpoint, using the symmetry of the random walk, we can argue that the y-coordinate should be equal to 0.

EXERCISE 2.5. By running the following script, we simulate trajectories and calculate the number of those that hit the barrier. The plot is given below.

```
Nhits<- c()

#specifying seed
set.seed(96770)

#defining walk as matrix
walk<- matrix(NA, nrow=1001, ncol=2)

#defining random steps
rstep<- matrix(c(1, 0, -1, 0, 0, 1, 0, -1), nrow=4, ncol=2, byrow=TRUE)

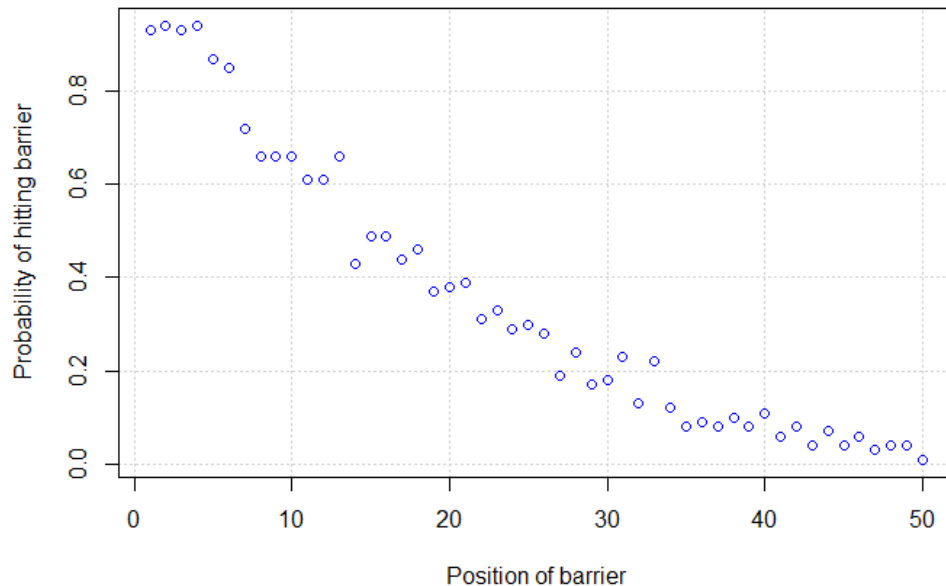
#varying the barrier value
for (barrier in 1:50) {
  nhits<- 0
  #simulating trajectories
  for (j in 1:100)
  {
    walk[1,]<- c(0,0) #setting initial value to the origin
    for (i in 2:1001)
    {
      walk[i,]<- walk[i-1,] + rstep[sample(1:4, size=1),]

      if (walk[i,1]==barrier) {
        nhits=nhits+1
        break
      }
    }
  }
  Nhits[barrier]=nhits
}
```

```
print(Nhits)
```

```
[1] 93 94 93 94 87 85 72 66 66 66 61 61
[13] 66 43 49 49 44 46 37 38 39 31 33 29
[25] 30 28 19 24 17 18 23 13 22 12  8  9
[37]  8 10  8 11  6  8  4  7  4  6  3  4
[49]  4  1
```

```
plot(1:50, Nhits/100, col="blue", xlab="Position of barrier", ylab="Probability of
hitting barrier", panel.first=grid())
```



We see that as the barrier value increases from 1 to 50, the estimated probability of hitting this barrier decreases from 0.93 to 0.01, in a slightly curvilinear (convex downward) manner.

EXERCISE 2.6. The lines of code given below terminate each trajectory if it reaches a side of the square. The total number of steps required is recorded for each trajectory. The average value is computed at the end.

```
walk<- data.frame()
nsteps<- c()

set.seed(37440)

#defining random steps
rstep<- matrix(c(1, 0, -1, 0, 0, 1, 0, -1), nrow=4, ncol=2, byrow=TRUE)

#simulating trajectories
for (j in 1:1000)
{
  walk[1,1]<- 0
  walk[1,2]<- 0
  i<- 2

  repeat{
    walk[i,]<- walk[i-1,] + rstep[sample(1:4, size=1),]
```

```

    if (walk[i,1]==10 | walk[i,1]==-10 | walk[i,2]==10 | walk[i,2]==-10)
    { break }
    else i=i+1
    }

    nsteps[j]=i
}

mean(nsteps)

120.898

```

Hence, the average number of steps it takes for the random walk to reach the square is estimated as 120.898 steps.

EXERCISE 2.7. (a) Conditioning on the outcome of the first step, we see that the probability \mathbf{P}_i solves the recurrence relation $\mathbf{P}_i = p\mathbf{P}_{i+1} + q\mathbf{P}_{i-1}$ with the border constraints $\mathbf{P}_A = 0$ and $\mathbf{P}_B = 1$. Assuming first that $\frac{q}{p} \neq 1$, we look for the solution in the form $\mathbf{P}_i = c\left(\frac{q}{p}\right)^i + d$ where c and d are some constants that can be found from the boundary conditions: $\mathbf{P}_A = 0 = c\left(\frac{q}{p}\right)^A + d$ and $\mathbf{P}_B = 1 = c\left(\frac{q}{p}\right)^B + d$. From here, $c = -\frac{1}{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^B}$ and $d = \frac{\left(\frac{q}{p}\right)^A}{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^B}$, and thus, $\mathbf{P}_i = \frac{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^i}{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^B}$.

Now assume $\frac{q}{p} = 1$. We look for the solution of the recurrence relation in the form $\mathbf{P}_i = ci + d$. Again, from the boundary conditions, $\mathbf{P}_A = 0 = cA + d$ and $\mathbf{P}_B = 1 = cB + d$. Hence, $c = \frac{1}{B-A}$ and $d = -\frac{A}{B-A}$, leading to $\mathbf{P}_i = \frac{i-A}{B-A}$.

(b) By conditioning on the first step, we see right away that the expectation satisfies the recurrence relation $\mathbf{E}_i = p\mathbf{E}_{i+1} + q\mathbf{E}_{i-1} + 1$ with the boundary conditions $\mathbf{E}_A = \mathbf{E}_B = 0$. Because of the additive constant term, this equation is referred to as a non-homogeneous relation and the general solution is sought in the form $\mathbf{E}_i = c\left(\frac{q}{p}\right)^i + d + \frac{i}{q-p}$, if $\frac{q}{p} \neq 1$, and $\mathbf{E}_i = ci + d - i^2$, if $\frac{q}{p} = 1$. The constants c and d are found from the boundary conditions. In the former case,

$$\text{they satisfy } \mathbf{E}_A = 0 = c\left(\frac{q}{p}\right)^A + d + \frac{A}{q-p} \text{ and } \mathbf{E}_B = 0 = c\left(\frac{q}{p}\right)^B + d + \frac{B}{q-p}. \text{ Whence,}$$

$$c = \frac{B-A}{q-p} \cdot \frac{1}{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^B},$$

and

$$d = -\frac{B-A}{q-p} \cdot \frac{\left(\frac{q}{p}\right)^B}{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^B} - \frac{B}{q-p},$$

resulting in

$$E_i = \frac{B - A}{q - p} \cdot \frac{\left(\frac{q}{p}\right)^i - \left(\frac{q}{p}\right)^B}{\left(\frac{q}{p}\right)^A - \left(\frac{q}{p}\right)^B} - \frac{B - i}{q - p}.$$

In the latter case, c and d solve $E_A = 0 = cA + d - A^2$ and $E_B = 0 = cB + d - B^2$. From here, $c = A + B$ and $d = -AB$. Thus, $E_i = (A + B)i - AB - i^2 = (B - i)(i - A)$.

(c) We use the formulas derived above with $p = 0.47$, $q = 0.53$, $A = 10$, $i = 40$, and $B = 80$. We obtain

$$P_{40} = \frac{\left(\frac{0.53}{0.47}\right)^{10} - \left(\frac{0.53}{0.47}\right)^{40}}{\left(\frac{0.53}{0.47}\right)^{10} - \left(\frac{0.53}{0.47}\right)^{80}} = 0.007962,$$

and

$$E_{40} = \frac{80 - 10}{0.53 - 0.47} \cdot \frac{\left(\frac{0.53}{0.47}\right)^{40} - \left(\frac{0.53}{0.47}\right)^{80}}{\left(\frac{0.53}{0.47}\right)^{10} - \left(\frac{0.53}{0.47}\right)^{80}} - \frac{80 - 40}{0.53 - 0.47} = 490.7115.$$

The probability of doubling the fortune in this rigged game is very small (about 0.008), and the gambler will play, on average, about 491 games before he walks out of the casino.

Below we give the code that simulates 10,000 trajectories and computes the proportion of them that ended in \$80 (as opposed to \$10) and averages the number of games in each trajectory.

```
#specifying parameters
p<- 0.47
ntraj<- 10000

#setting seed number
set.seed(314159)

#defining walk as matrix
walk<- data.frame(NULL)

#setting counters
n80<- 0
n10<- 0
ngames<- 0

#simulating trajectories
for (j in 1:ntraj) {
  walk[1,j]<- 40
  k<- 2
  repeat {
    walk[k,j]<- ifelse(runif(1)<p, walk[k-1,j]+1, walk[k-1,j]-1)
    if(walk[k,j]==80) {
      n80<- n80+1
      break
    }
    if(walk[k,j]==10) {
      n10<- n10+1
      break
    }
  }
}
```

```

    k<- k+1
    ngames<- ngames+1
  }
}

print(prop.n80<- n80/ntraj)

0.0084

print(avg.ngames<- ngames/ntraj)

488.4926

```

EXERCISE 2.8. The student's visit to the museum can be modeled as a random walk on a graph with the state space $S = \{Exit, A, B, C, D, E, F\}$, and the transition probability matrix \mathbf{P}

	<i>Exit</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>
<i>Exit</i>	1	0	0	0	0	0	0
<i>A</i>	1/3	0	1/3	0	1/3	0	0
<i>B</i>	0	1/2	0	1/2	0	0	0
<i>C</i>	0	0	1/3	0	1/3	1/3	0
<i>D</i>	0	1/2	0	1/2	0	0	0
<i>E</i>	0	0	0	1/2	0	0	1/2
<i>F</i>	0	0	0	0	0	1	0

We assume that at the beginning of the walk, the student enters the museum and finds himself in Room A. The expected number of transitions between the rooms until he reaches the exit is given by the formula

$$\begin{aligned}
 E(\# \text{ of transitions}) &= (0, 1, 0, 0, 0, 0, 0) \left((1)(\mathbf{P}) + (2)(\mathbf{P}^2 - \mathbf{P}) + (3)(\mathbf{P}^3 - \mathbf{P}^2) + (4)(\mathbf{P}^4 - \mathbf{P}^3) \right. \\
 &\quad \left. + \dots \right) (1, 0, 0, 0, 0, 0, 0)^{-1}
 \end{aligned}$$

We submit the following R code that approximates this sum. The convergence is achieved with 146 terms.

```

#specifying the transition probability matrix
tm<- matrix(c(1,0,0,0,0,0,0,1/3,0,1/3,0,1/3,0,0,0,1/2,0,1/2,0,0,0,0,0,0,
1/3,0,1/3,1/3,0,0,0,1/2,0,1/2,0,0,0,0,0,0,1/2,0,0,1/2,0,0,0,0,0,1,0),
nrow=7, ncol=7, byrow=TRUE)

#setting counter
ntrans<- 0

#computing expected number of transitions
p<- matrix(NA, nrow=146, ncol=7)
p[1,]<- c(0,1,0,0,0,0,0)

for (i in 2:146) {
  p[i,]<- p[i-1,]%*%tm
  ntrans<- ntrans+(i-1)*(p[i,1]-p[i-1,1])
}

```



```
print(ntrans)
```

12.94225

Thus, the student will make, on average, 12.94225 transitions between the rooms. Since he spends 30 minutes in each room, the total average length of visit will be $(30)(12.94225) = 388.2675$ minutes (or 6 hours and 28.3 minutes). On an “average” visit, he will be done before the museum closes for the day.

CHAPTER 3

EXERCISE 3.1. We use the independence and stationarity of increments of a Poisson process to derive the expression for the joint probability distribution. We write

$$\begin{aligned}
 P(N(s) = m, N(t) = n) &= P(N(t) - N(s) = n - m, N(s) = m) \\
 &= P(N(t) - N(s) = n - m)P(N(s) = m) = P(N(t - s) = n - m)P(N(s) = m) \\
 &= \frac{(\lambda(t - s))^{n-m}}{(n - m)!} e^{-\lambda(t-s)} \cdot \frac{(\lambda s)^m}{m!} e^{-\lambda s} = \frac{(t - s)^{n-m} s^m}{(n - m)! m!} \lambda^n e^{-\lambda t} \\
 &= \binom{n}{m} \left(\frac{s}{t}\right)^m \left(1 - \frac{s}{t}\right)^{n-m} \frac{\lambda^n}{n!} e^{-\lambda t}.
 \end{aligned}$$

EXERCISE 3.2. Assume that $s < t$. We compute the covariance function, using the independence and stationarity of the increments. We have

$$\begin{aligned}
 \text{Cov}(N(s), N(t)) &= E[N(s)N(t)] - E[N(s)]E[N(t)] \\
 &= E[(N(t) - N(s) + N(s))(N(s))] - E[N(s)]E[N(t)] \\
 &= E[(N(t) - N(s))N(s)] + E[N(s)]^2 - E[N(s)]E[N(t)] \\
 &= E[N(t) - N(s)]E[N(s)] + \text{Var}[N(s)] + [E(N(s))]^2 - E[N(s)]E[N(t)] \\
 &= E[N(t - s)]E[N(s)] + \text{Var}[N(s)] + [E(N(s))]^2 - E[N(s)]E[N(t)] \\
 &= \lambda(t - s)\lambda s + \lambda s + (\lambda s)^2 - \lambda s \lambda t = \lambda s.
 \end{aligned}$$

EXERCISE 3.3. (a) $P(N(5) = 16 \mid N(1) = 2, N(2) - N(1) = 3)$

$$\begin{aligned}
 &= \frac{P(N(5) - N(2) = 11, N(2) - N(1) = 3, N(1) = 2)}{P(N(2) - N(1) = 3, N(1) = 2)} \\
 &= \frac{P(N(3) = 11)P(N(1) = 3)P(N(1) = 2)}{P(N(1) = 3)P(N(1) = 2)} = P(N(3) = 11) = \frac{((5)(3))^{11}}{11!} e^{-(5)(3)} = 0.0663.
 \end{aligned}$$

(b) Since $E(S_{100}) = (100) \left(\frac{1}{5}\right) = 20$, the 100th claim is expected to be seen on the 20th business day, that is, on January 27th.

JANUARY					
Monday	2	9	16	23	30
Tuesday	3	10	17	24	31
Wednesday	4	11	18	25	
Thursday	5	12	19	26	
Friday	6	13	20	27	
Saturday	7	14	21	28	
Sunday	8	15	22	29	

EXERCISE 3.4. (a) Phone calls that result in sales occur with rate $(0.15)\left(\frac{60}{5}\right) = 1.8$ per hour.

Therefore, in the next two hours, there will be, on average $(2)(1.8)=3.6$ successful sales.

(b) The total number of phone calls is a Poisson process with a rate of $60/5=12$ per hour. Phone calls that result in a sale and those that don't form independent Poisson processes with rates 1.8 and 10.2 per hour, respectively. Therefore,

$$\begin{aligned} P(N(1) = 15, N_{\text{sale}}(1) = 5) &= P(N_{\text{sale}}(1) = 5, N_{\text{no sale}}(1) = 10) \\ &= P(N_{\text{sale}}(1) = 5)P(N_{\text{no sale}}(1) = 10) = \frac{(1.8)^5}{5!} e^{-1.8} \frac{(10.2)^{10}}{10!} e^{-10.2} = 0.00325. \end{aligned}$$

$$(c) P(N(4) = 10 | N(1) = 3) = P(N(4) - N(1) = 7) = P(N(3) = 7) = \frac{((1.8)(3))^7}{7!} e^{-(1.8)(3)} = 0.119987.$$

EXERCISE 3.5. (a) $N_1(t)$ and $N_2(t)$ are splitted Poisson processes with the means

$$\begin{aligned} E(N_1(t)) &= \lambda \int_0^t P(\text{disease is contracted at time } s, \text{ symptoms show by time } t) ds \\ &= \lambda \int_0^t F(t-s) ds = \{u = t-s\} = \lambda \int_0^t F(u) du, \end{aligned}$$

and

$$\begin{aligned} E(N_2(t)) &= \lambda \int_0^t P(\text{disease is contracted at time } s, \text{ no symptoms show by time } t) ds \\ &= \lambda \int_0^t (1 - F(t-s)) ds = \{u = t-s\} = \lambda \int_0^t (1 - F(u)) du. \end{aligned}$$

(b) Suppose by a fixed time t , $\hat{E}(N_1(t))$ individuals are observed who show symptoms of a disease. From here, we can estimate the rate of contracting the disease as $\hat{\lambda} = \frac{\hat{E}(N_1(t))}{\int_0^t F(u) du}$. Plugging this into the expression for the expected value of $N_2(t)$, we can calculate the estimated number of individuals infected but not yet showing symptoms by time t as

$$\hat{E}(N_2(t)) = \frac{\hat{E}(N_1(t)) \int_0^t (1 - F(u)) du}{\int_0^t F(u) du}.$$

(c) Suppose the incubation period until symptoms show is an exponentially distributed random variable with a mean of 2 days. Thus, $F(u) = 1 - e^{-u/2}$, $u \geq 0$. Given that $\hat{E}(N_1(10)) = 1000$, we estimate the number of individuals who are infected but haven't shown the symptoms yet as

$$\hat{E}(N_2(10)) = \frac{1000 \int_0^{10} e^{-\frac{u}{2}} du}{\int_0^{10} (1 - e^{-\frac{u}{2}}) du} = \frac{(1000)(2) \left(1 - e^{-\frac{10}{2}}\right)}{10 - (2) \left(1 - e^{-\frac{10}{2}}\right)} = 247.8979,$$

or about 248 individuals.

EXERCISE 3.6. (a) Let $N(t)$ denote the number of high road surface distress areas on a t -mile stretch of the road. It is a Poisson process with a rate $\lambda = 2.8$. So, $E(N(10)) = (2.8)(10) = 28$.

(b) The code below simulates 30 distances between distressed surface areas. These distances are independent and exponentially distributed with mean $\frac{1}{2.8} = 0.357143$ miles.

```
#specifying parameters
lambda<- 2.8
Nareas<- 30

#defining states
N<- 0:Nareas

#setting distance as vector
dist<- c()

#setting initial value for distance
dist[1]<- 0

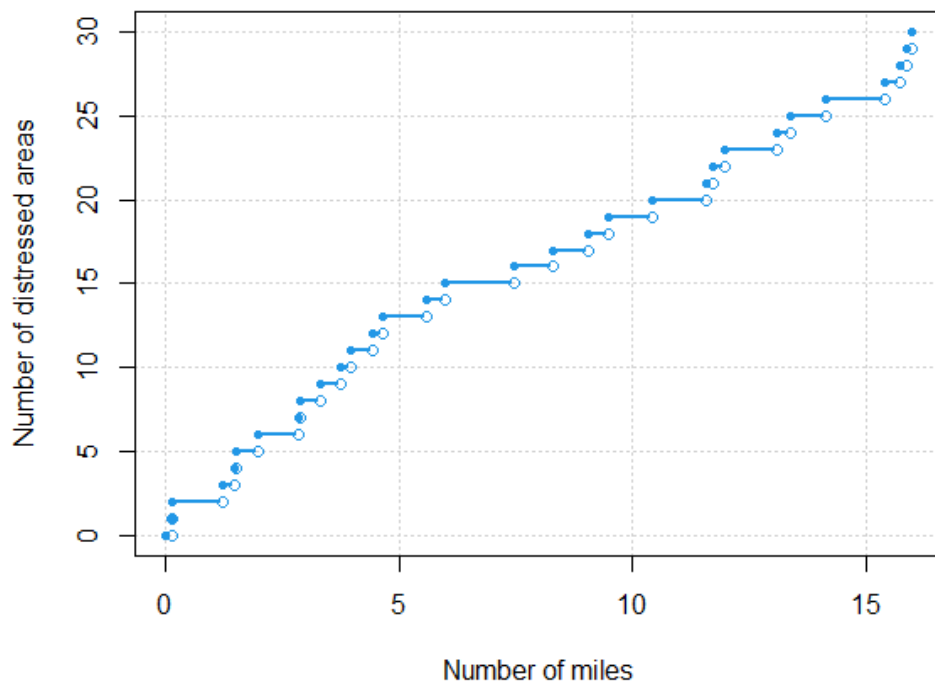
#specifying seed
set.seed(777754)

for (i in 2:(Nareas+1))
  dist[i]<- dist[i-1] + round((-1/lambda)*log(runif(1)),2)

#plotting trajectory
plot(dist, N, type="n", xlab="Number of miles", ylab="Number of distressed areas",
panel.first = grid())

segments(dist[-length(dist)],N[-length(dist)], dist[-1]-0.07, N[-length(dist)],
lwd=2, col=4)

points(dist, N, pch=20, col=4)
points(dist[-1],N[-length(dist)],pch=1, col=4)
```



In this simulation, the total length of the road that contains 30 distressed surface areas is the last value in the vector `dist`, that is, 15.99 miles.

```
dist[length(dist)]
```

15.99

(c) Given that $N(10) = 30$, the distances between distressed surface areas are distributed as order statistics of the uniform distribution on the interval $(0,10)$. The R code below simulates the locations of those areas.

```
#specifying parameters
D<- 10
Nareas<- 30

#specifying seed
set.seed(87998)

#defining states
N<- 0:Nareas

#generating N standard uniforms
u<- c()
u[1]<- 0

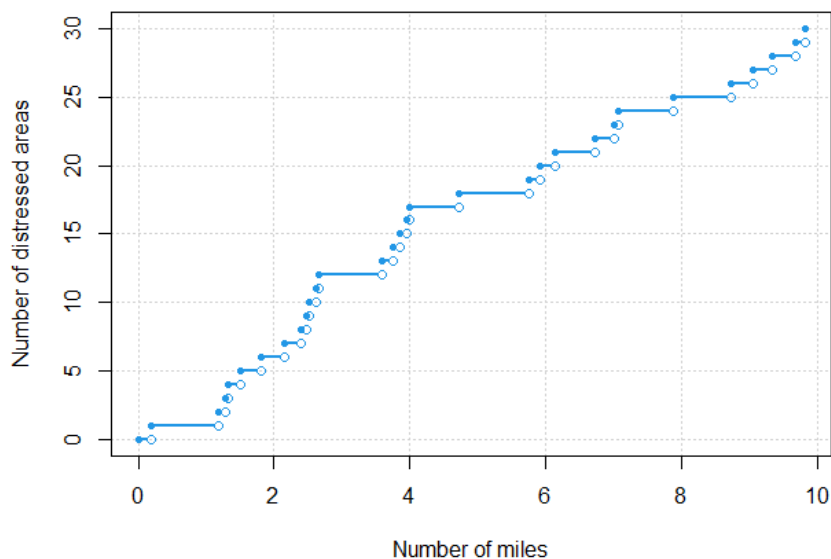
for(i in 2:(Nareas+1))
  u[i]<- runif(1)

#computing event distances
dist<- D*sort(u)

#plotting trajectory
plot(dist, N, type = "n", xlab="Number of miles", ylab="Number of distressed
areas", panel.first = grid())

segments(dist[-length(dist)],N[-length(dist)], dist[-1]-0.07, N[-length(dist)],
lwd=2, col=4)

points(dist, N, pch=20, col=4)
points(dist[-1],N[-length(dist)],pch=1, col=4)
```



EXERCISE 3.7. For the data set on significant volcanic eruptions between 1920 and 2020, the code below calculates interarrival times, plots a histogram, and conducts the goodness-of-fit test. The p-value for the test is larger than 0.05, indicating that the Poisson process models the data well.

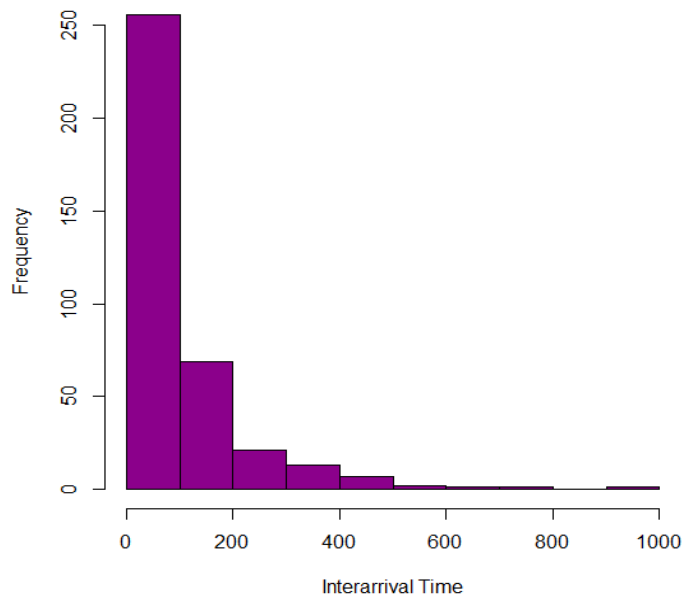
```
volcanoes.data<- read.csv(file="./volcanoesdata.csv", header=TRUE, sep=",")

#creating date-time variable
datetime<- as.POSIXct(paste(as.Date(volcanoes.data$DATE), volcanoes.data$TIME))

#computing lag
datetime.lag<- c(0,head(datetime, -1))

#computing interarrival times (in hours)
int<- (as.numeric(datetime)-as.numeric(datetime.lag))/(3600*24)
int<- int[-1] # removing first value

#plotting histogram
hist(int, main="", xlab="Interarrival Time", col="dark magenta")
```



```
#binning interarrival times
binned.int<- as.factor(ifelse(int<25,"1", ifelse(int>=25 & int<50,"2",
ifelse(int>=50 & int<100,"3", ifelse(int>=100 & int<150,"4",ifelse(int>=150 &
int<200,"5", ifelse(int>=200 & int<250,"6", "7"))))))))

#computing observed frequencies
obs<- table(binned.int)

#estimating mean for exponential distribution
mean.est<- mean(int)

#computing expected frequencies
exp<- c(1:7)
exp[1]<- length(int)*(1-exp(-25/mean.est))
exp[2]<- length(int)*(exp(-25/mean.est)-exp(-50/mean.est))
exp[3]<- length(int)*(exp(-50/mean.est)-exp(-100/mean.est))
exp[4]<- length(int)*(exp(-100/mean.est)-exp(-150/mean.est))
exp[5]<- length(int)*(exp(-150/mean.est)-exp(-200/mean.est))
```

```

exp[6]<- length(int)*(exp(-200/mean.est)-exp(-250/mean.est))
exp[7]<- length(int)*exp(-250/mean.est)

obs

1      2      3      4      5      6      7
96     64     94     42     29     13     33

round(exp,1)

83.3  64.6  88.9  53.5  32.2  19.3  29.2

#computing chi-squared statistic
print(chi.sq<- sum((obs-exp)^2/exp))

7.581284

#computing p-value
print(p.value<- 1-pchisq(chi.sq, df=5))

0.1808718

```

EXERCISE 3.8. (a) Team A scores as a Poisson process with a rate $\lambda_A = (0.25 + 0.40 + 0.20)(0.5) = (0.85)(0.5) = 0.425$ per minute. Team B scores as a Poisson process with a rate $\lambda_B = (0.25 + 0.50 + 0.15)(0.4) = (0.9)(0.4) = 0.36$ per minute. Hence,

$$E(\text{time until a team scores}) = \frac{1}{\lambda_A + \lambda_B} = \frac{1}{0.425 + 0.36} = 1.273885 \text{ minutes.}$$

$$(b) E(\text{time until team A scores}) = \frac{1}{\lambda_A} = \frac{1}{0.425} = 2.352941 \text{ minutes.}$$

$$E(\text{time until team B scores}) = \frac{1}{\lambda_B} = \frac{1}{0.36} = 2.777778 \text{ minutes.}$$

$$(c) P(\text{team A scores before team B}) = \frac{\lambda_A}{\lambda_A + \lambda_B} = \frac{0.425}{0.425 + 0.36} = 0.541401,$$

$$P(\text{team B scores before team A}) = 1 - 0.541401 = 0.458599.$$

(d) For team A, 1-pointers occur as an independent Poisson process with rate $(0.25)(0.5) = 0.125$ per minute; 2-pointers occur as an independent Poisson process with rate $(0.4)(0.5) = 0.2$ per minute; and 3-pointers occur as a Poisson process with rate $(0.20)(0.5) = 0.1$ per minute. For team B, the respective rates are $(0.25)(0.4) = 0.1$ per minute, $(0.5)(0.4) = 0.2$ per minute, and $(0.15)(0.4) = 0.06$ per minute. Therefore,

$$P(\text{teams score same \# of 1-pointers, 2-pointers, and 3-pointers})$$

$$\begin{aligned}
&= P(\text{same \# of 1-pointers})P(\text{same \# of 2-pointers})P(\text{same \# of 3-pointers}) \\
&= \left[e^{-(0.125+0.1)(48)} \sum_{n=0}^{\infty} \frac{((0.125)(0.1)(48)^2)^n}{(n!)^2} \right] \left[e^{-(0.2+0.2)(48)} \sum_{n=0}^{\infty} \frac{((0.2)(0.2)(48)^2)^n}{(n!)^2} \right] \\
&\quad \times \left[e^{-(0.1+0.06)(48)} \sum_{n=0}^{\infty} \frac{((0.1)(0.06)(48)^2)^n}{(n!)^2} \right] = 0.0012.
\end{aligned}$$

#computing probability of same number of 1-pointers, 2-pointers, and 3-pointers

```
sum1<- 0
for(n in 0:16)
  sum1<- sum1+(0.125*0.1*48^2)^n/(factorial(n))^2

print(p1<- sum1*exp(-(0.125+0.1)*48))
```

0.1152988

```
sum2<- 0
for(n in 0:23)
  sum2<- sum2+(0.2*0.2*48^2)^n/(factorial(n))^2

print(p2<- sum2*exp(-(0.2+0.2)*48))
```

0.09165684

```
sum3<- 0
for(n in 0:12)
  sum3<- sum3+(0.1*0.06*48^2)^n/(factorial(n))^2

print(p3<- sum3*exp(-(0.1+0.06)*48))
```

0.1167362

```
p1*p2*p3
```

0.001233658

EXERCISE 3.9. (a) The spider will need $\tau = 30$ minutes = 0.5 hours to reach the top. The rate of rain is $\lambda = 2$ per hour. Denote by T the total time it takes the spider to reach the top. The expected value of T is $E(T) = \frac{1}{2}(e^{(0.5)(2)} - 1) = 0.859141$ hours or 51.5 minutes.

(b) Let N denote the number of times the spider will be washed down before it reaches the top. Then, $E(N) = e^{(0.5)(2)} - 1 = 1.718282$.

CHAPTER 4

EXERCISE 4.1. (a) The number of broken calculators can be modeled according to a nonhomogeneous Poisson process $\{N(t), t \geq 0\}$ with the intensity rate function

$$\lambda(t) = \begin{cases} 3, & \text{if } 0 \leq t \leq 3, \\ 2t - 3, & \text{if } 3 \leq t \leq 10. \end{cases}$$

The integrated rate function is

$$\Lambda(t) = \int_0^t \lambda(u) du = \begin{cases} \int_0^t 3 du = 3t, & \text{if } 0 \leq t \leq 3, \\ 9 + \int_3^t (2u - 3) du = t^2 - 3t + 9, & \text{if } 3 \leq t \leq 10. \end{cases}$$

The probability mass function is

$$P(N(t) - N(s) = n) = \frac{(\Lambda(t) - \Lambda(s))^n}{n!} e^{-(\Lambda(t) - \Lambda(s))} = \frac{(t^2 - s^2 - 3(t - s))^n}{n!} e^{-(t^2 - s^2 - 3(t - s))}.$$

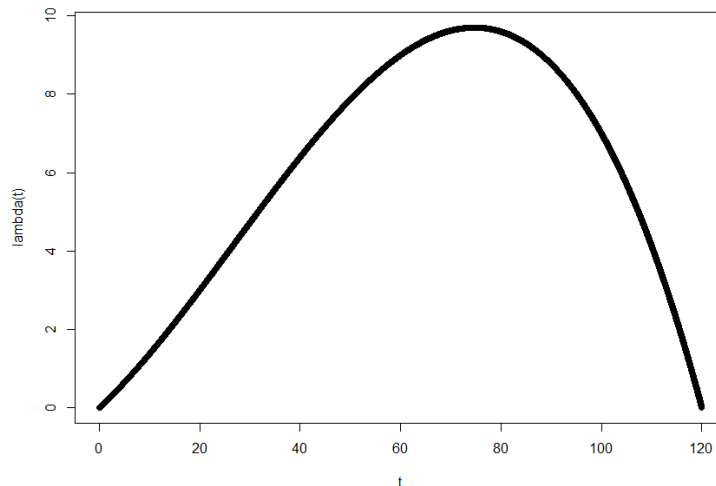
$$(b) P(N(8) - N(4) = 50) = \frac{(8^2 - 4^2 - 3(8 - 4))^{50}}{50!} e^{-(8^2 - 4^2 - 3(8 - 4))} = 0.004983.$$

$$(c) E(N(10) - N(2)) = \Lambda(10) - \Lambda(2) = 10^2 - (3)(10) + 9 - (3)(2) = 73.$$

EXERCISE 4.2. (a) Below is the R code and plot of the intensity function $\lambda(t) = -0.000025 t^3 + 0.002 t^2 + 0.12t$ against t on the interval $[0, 120]$.

```
lambda<- function(t) -0.000025*t^3+0.002*t^2+0.12*t
t<- seq(0, 120, by = 0.01)

plot(t, lambda(t))
```

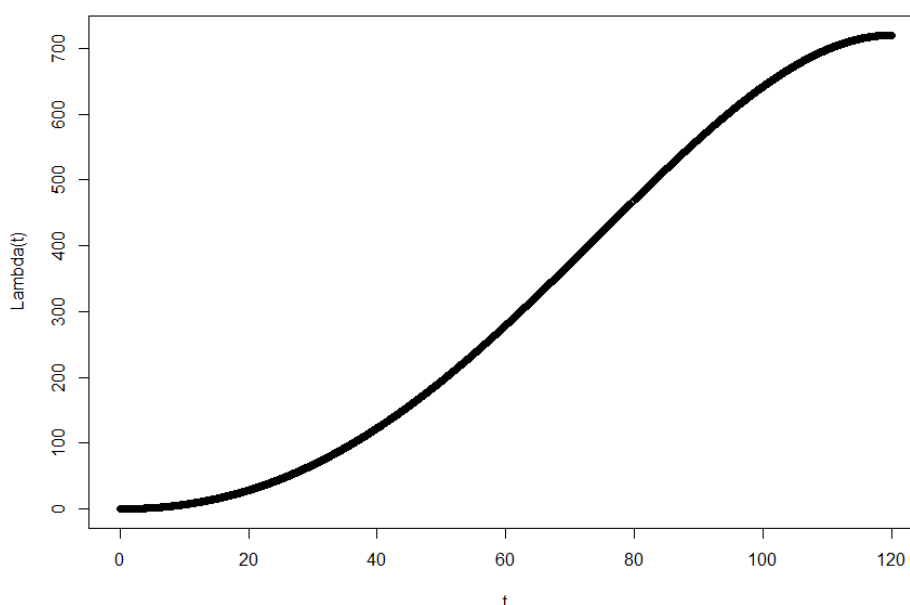


The intensity function looks like a skewed upside-down parabola that is equal to 0 at $t = 0$ and $t = 120$. It achieves the maximum at time t that solves the equation $\lambda'(t) = 0$. Therefore, t solves the quadratic equation $(-0.000025)(3)t^2 + (0.002)(2)t + 0.12 = 0$, which simplifies to $-0.00075t^2 + 0.4t + 12 = 0$. We need the solution that lies between 0 and 120, therefore, $t = \frac{0.4 + \sqrt{0.52}}{0.015} = 74.74068$ days. Thus the peak intensity rate occurs 74.74 days into the fire season, and the maximum number of fires per day is $-0.000025(74.74068)^3 + 0.002(74.74068)^2 + 0.12(74.74068) = 9.703286$, or roughly 9.7 fires per day.

(b) We write $\Lambda(t) = \int_0^t \lambda(u) du = \int_0^t (-0.000025 u^3 + 0.002 u^2 + 0.12u) du = -0.00000625t^4 + 0.0006667t^3 + 0.06t^2, 0 \leq t \leq 120$. The R code and the graph are given below.

```
Lambda<- function(t) -0.00000625*t^4+0.0006667*t^3+0.06*t^2
t<- seq(0, 120, by = 0.01)

plot(t, Lambda(t))
```



To find the average number of wildfires per season we compute $\Lambda(120) = -(0.00000625)(120)^4 + (0.0006667)(120)^3 + (0.06)(120)^2 = 720.0576$, so, on average, about 720 fires occur in this area every season.

(a) The middle 50% of the fire season falls between day 30 and day 90. The mean number of wildfires in this period is $\Lambda(90) - \Lambda(30) = -(0.00000625)(90)^4 + (0.0006667)(90)^3 + (0.06)(90)^2 - (-(0.00000625)(30)^4 + (0.0006667)(30)^3 + (0.06)(30)^2) = 561.9618 - 66.9384 = 495.0234$.

EXERCISE 4.3. (a) The integrated intensity function is

$$\Lambda(t) = \int_0^t \lambda(u) du = \int_0^t \frac{A}{\sqrt{u}} du = 2A\sqrt{t}, t \geq 0.$$

We know that $\Lambda(1) = 30$. Thus, $A = 15$.

(b) Given that the n th injury occurred at the time s_n , the conditional cumulative distribution function of the time until the next injury T_{n+1} , $n \geq 1$, is $F_{T_{n+1}|s_n}(t|s_n) = 1 - \exp\{-30(\sqrt{t+s_n} - \sqrt{s_n})\}$, $t \geq 0$. The cdf of T_1 is $F_{T_1}(t) = 1 - \exp\{-30\sqrt{t}\}$, $t \geq 0$. To simulate observations from this distribution, we first generate standard uniform random variables U_n , $n \geq 1$, and then sequentially solve $F_{T_{n+1}|s_n}(t|s_n) = U_{n+1}$, i.e., we solve for t the identities $1 - \exp\{-30(\sqrt{t+s_n} - \sqrt{s_n})\} = U_n$. Replacing $1 - U_n$ by U_n since both have the same standard uniform distribution, we obtain

$$S_1 = \left(-\frac{1}{30}\ln(U_n)\right)^2, \quad \text{and} \quad S_{n+1} = \left(\sqrt{S_n} - \frac{1}{30}\ln(U_n)\right)^2, \quad n \geq 1.$$

The code and the simulated trajectory follow.

```
#specifying parameters
ninjrs<- 100

#defining states
N<- 0:ninjrs

#defining times as vectors
time<- c()

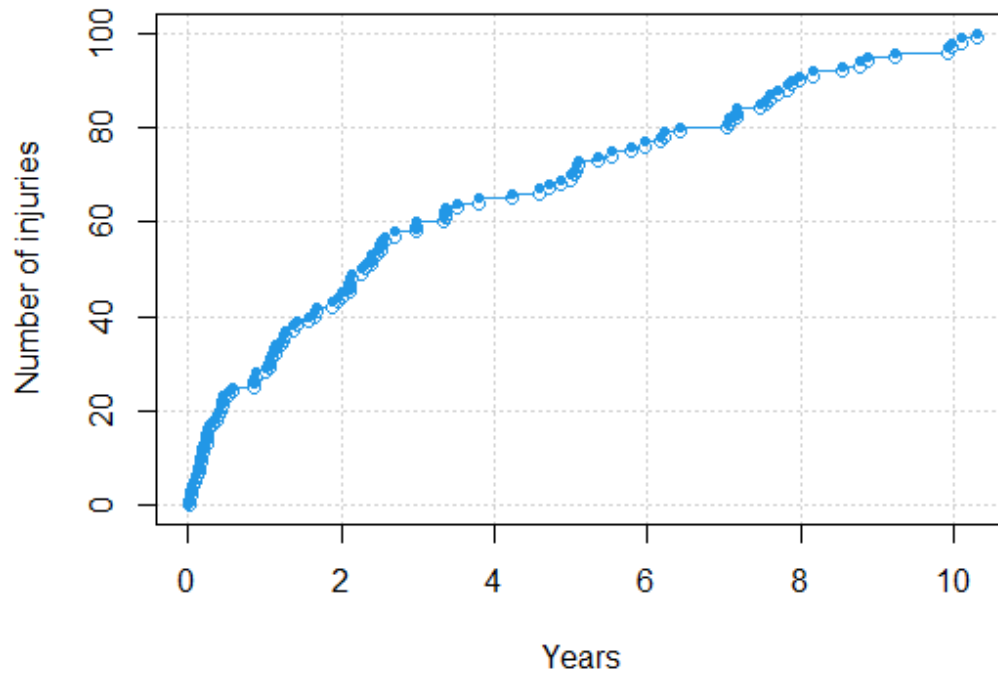
#specifying seed
set.seed(933541)

#computing event times
time[1]<- (-log(runif(1))/30)^2
for(i in 2:(ninjrs+1))
  time[i]<- (sqrt(time[i-1])-log(runif(1))/30)^2

#plotting simulated trajectory
plot(time, N, type="n", xlab="Years", ylab="Number of injuries",
panel.first=grid())

segments(time[-length(time)], N[-length(time)], time[-1]-0.07,
N[-length(time)], col=4)

points(time, N, pch=20, col=4)
points(time[-1], N[-length(time)], pch=1, col=4)
```



```
time[201]
```

```
9.870481
```

The time range of the trajectory is $[0, 9.870481]$.

(c) We are given that $S_{100} = 12.25$ years. The times of injuries S_1, S_2, \dots, S_{99} are then represented by order statistics from the distribution with the cumulative distribution function $\frac{\Lambda(s)}{\Lambda(S_{100})} = \frac{30\sqrt{s}}{30\sqrt{S_{100}}} = \sqrt{s/S_{100}}$. The code that simulates a trajectory and the graph follow.

```
t<- 12.25
Lambda<- 30*sqrt(t)

#generating N(t)
set.seed(1133664)
ninjrs<- rpois(1,Lambda)

#defining states
N<- 0:ninjrs

#generating standard uniforms
u<- c()
u[1]<- 0
for(i in 2:(ninjrs+1))
u[i]<- runif(1)

#computing event times
time<- round(t*sort(u)^2,4)

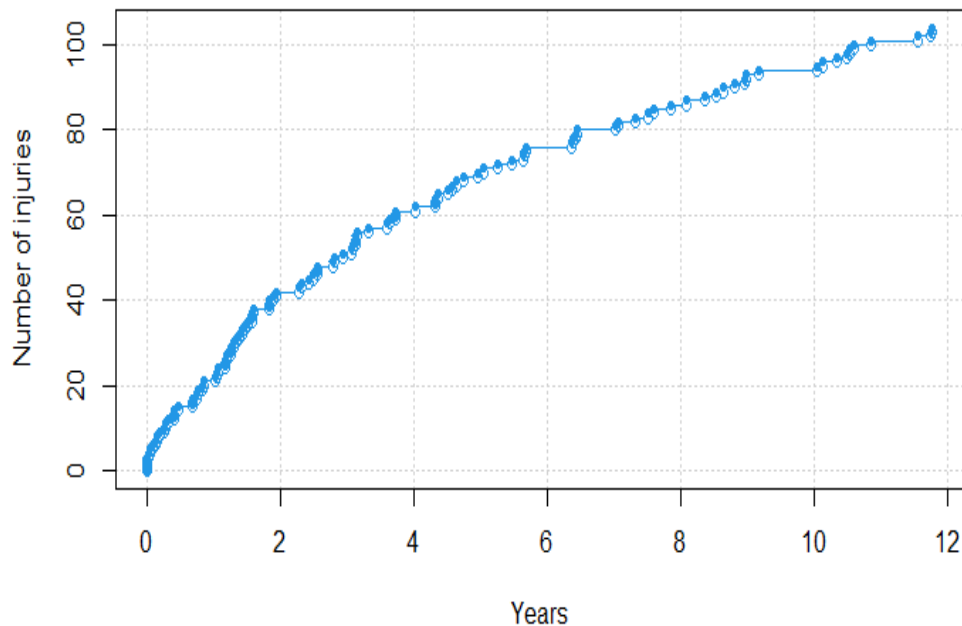
#plotting simulated trajectory
plot(time, N, type="n", xlab="Years", ylab="Number of injuries",
panel.first=grid())
```

```

segments(time[-length(time)], N[-length(time)], time[-1]-0.07,
N[-length(time)], col=4)

points(time, N, pch=20, col=4)
points(time[-1], N[-length(time)], pch=1, col=4)

```



EXERCISE 4.4. Below are the code and plot of the simulated trajectory. We are using the thinning simulation method where we uniformly dominate the intensity rate function by 20.

```

#specifying parameters
lambda<- function(s) 10+10*cos(2*pi*s)
lambda.star<- function(s) 20
Lambda.star<- function(s) 20*s

#specifying seed
set.seed(2866514)

#generating N(t)
njumps<- rpois(1, Lambda.star(10))

#generating N(t) standard uniforms
u<- c()
u[1]<- 0

for(i in 2:(njumps+1))
  u[i]<- runif(1)

#computing event times
time.star<- 10*sort(u)

#thinning event times
accepted<- c()
time<- c()
accepted[1]<- 1
time[1]<- 0

for (i in 2:(njumps+1)) {
  if (runif(1)<= lambda(time.star[i])/lambda.star(time.star[i]))

```

```

    accepted[i]=1 else accepted[i]=0
  }

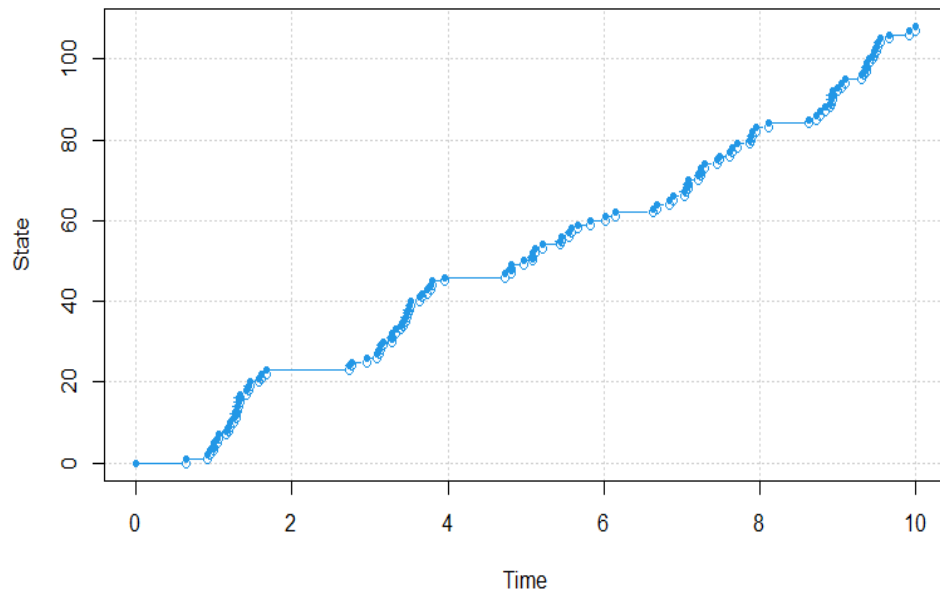
time<- time.star[-which(accepted==0)]
N<- 0:(length(time)-1)

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State", panel.first = grid())

segments(time[-length(time)],N[-length(time)], time[-1]-0.07,
          N[-length(time)], col=4)

points(time, N, ylim=c(0,120), pch=20, col=4)
points(time[-1],N[-length(time)],pch=1, col=4)

```



EXERCISE 4.5. In the process of radioactive decay, photons are emitted according to a nonhomogeneous Poisson process with the intensity rate $\lambda(t) = 100e^{-0.5t}$, $t \geq 0$. The integrated intensity rate function is $\Lambda(t) = \int_0^t \lambda(s)ds = 200(1 - e^{-0.5t})$, $t \geq 0$.

METHOD 1 (EXPONENTIAL INTERARRIVALS). To simulate event times, we solve the recurrence equations $1 - e^{-\Lambda(S_1)} = U_1$ and $1 - e^{-(\Lambda(S_{n+1}) - \Lambda(S_n))} = U_{n+1}$, $n \geq 1$. Equivalently, we can replace $1 - U_1$ by U_1 and solve $e^{-\Lambda(S_1)} = U_1$ and $e^{-(\Lambda(S_{n+1}) - \Lambda(S_n))} = U_{n+1}$, $n \geq 1$. In this example, the equations are: $e^{-200(1 - e^{-0.5S_1})} = U_1$ and $e^{-200(e^{-0.5S_n} - e^{-0.5S_{n+1}})} = U_{n+1}$, $n \geq 1$. Solving, we get

$$S_1 = -2\ln\left(1 + \left(\frac{1}{200}\right)\ln(U_1)\right) \text{ and } S_{n+1} = -2\ln\left(e^{-0.5S_n} + \left(\frac{1}{200}\right)\ln(U_{n+1})\right), n \geq 1.$$

The code below simulates a trajectory with 20 events.

```

#specifying parameters
njumps<- 20

#defining states
N<- 0:njumps

#defining times as vectors

```

```

time<- c()

#specifying seed
set.seed(40556002)

#generating standard uniforms
u<- c()
for(i in 1:njumps)
u[i]<- runif(1)

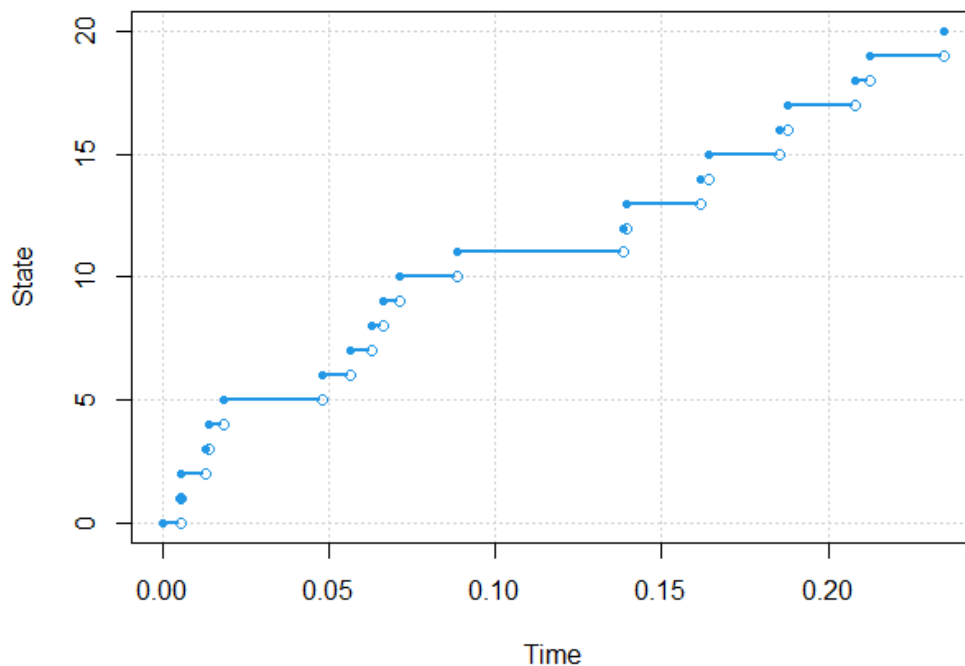
#computing event times
time[1]<- 0
time[2]<- -2*log(1+(1/200)*log(u[1]))

for(i in 3:(njumps+1)) {
time[i]<- -2*log(exp(-0.5*time[i-1])+(1/200)*log(u[i-1]))
}

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State", panel.first = grid())
segments(time[-length(time)],N[-length(time)], time[-1]-0.001, N[-length(time)]],
lwd=2, col=4)

points(time, N, ylim=c(0,120), pch=20, col=4)
points(time[-1],N[-length(time)], pch=1, col=4)

```



METHOD 2 (UNIFORM ORDER STATISTICS). In this method, an event time S is found as the solution of the equation $\frac{\Lambda(S)}{\Lambda(0.25)} = U$ where by U we denote a standard uniform random variable from an ordered sample. The equation takes the form: $\frac{200(1-e^{-0.5S})}{200(1-e^{-(0.5)(0.25)})} = U$. The solution is $S = -2 \ln(1 - (1 - e^{-(0.5)(0.25)})U)$. The code and graphical output are presented below.

```

#specifying parameters
t<- 0.25
Lambda<- 200*(1-exp(-0.5*t))

```

```

#specifying seed
set.seed(492231)

#generating N(t)
njumps<- rpois(1,Lambda)

#defining states
N<- 0:njumps

#generating N(t) standard uniforms
u<- c()
u[1]<- 0

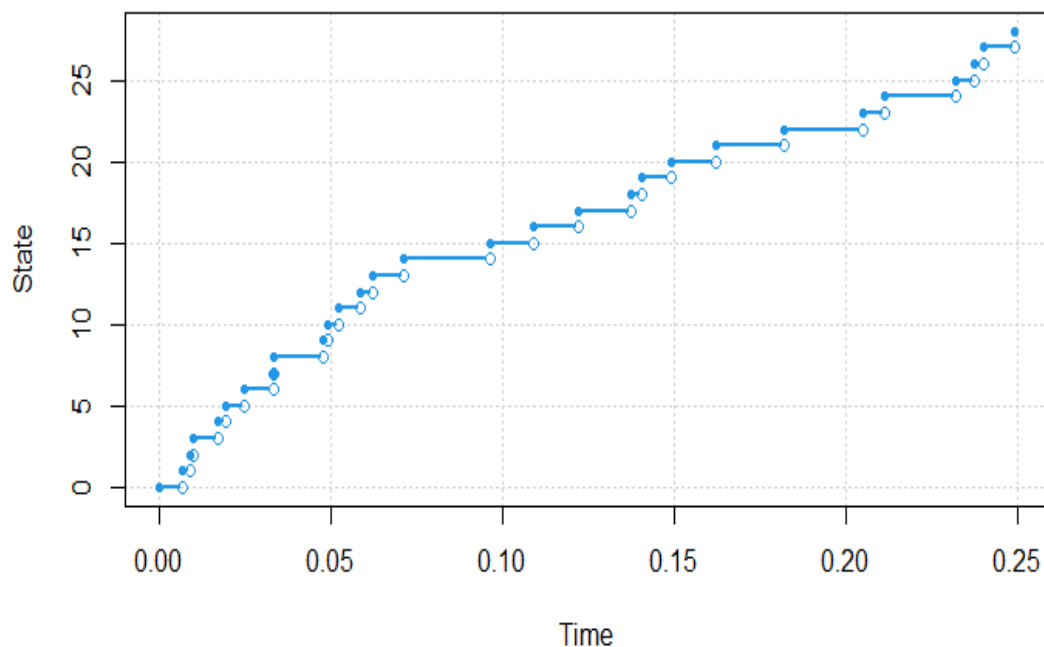
for(i in 2:(njumps+1))
  u[i]<- runif(1)

#computing event times
time<- -2*log(1-(1-exp(-0.5*t))*sort(u))

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State", panel.first = grid())
segments(time[-length(time)],N[-length(time)], time[-1]-0.001, N[-length(time)],
lwd=2, col=4)

points(time, N, ylim=c(0,120), pch=20, col=4)
points(time[-1],N[-length(time)], pch=1, col=4)

```



METHOD 3 (THINNING). We bound the intensity rate function $\lambda(t) = 100e^{-0.5t}$, $t \geq 0$, uniformly by 100. The code below simulates event times of a homogeneous Poisson process and then applies the algorithm of the thinning method to select only those event times that belong to the nonhomogeneous Poisson process. The plot follows.

```

#specifying parameters
lambda<- function(s) 100*exp(-0.5*s)
lambda.star<- function(s) 100
Lambda.star<- function(s) 100*s

#specifying seed

```



```

set.seed(2866514)

#generating N(t)
njumps<- rpois(1, Lambda.star(0.25))

#generating N(t) standard uniforms
u<- c()
u[1]<- 0

for(i in 2:(njumps+1))
  u[i]<- runif(1)

#computing event times
time.star<- 0.25*sort(u)

#thinning event times
accepted<- c()
time<- c()
accepted[1]<- 1
time[1]<- 0

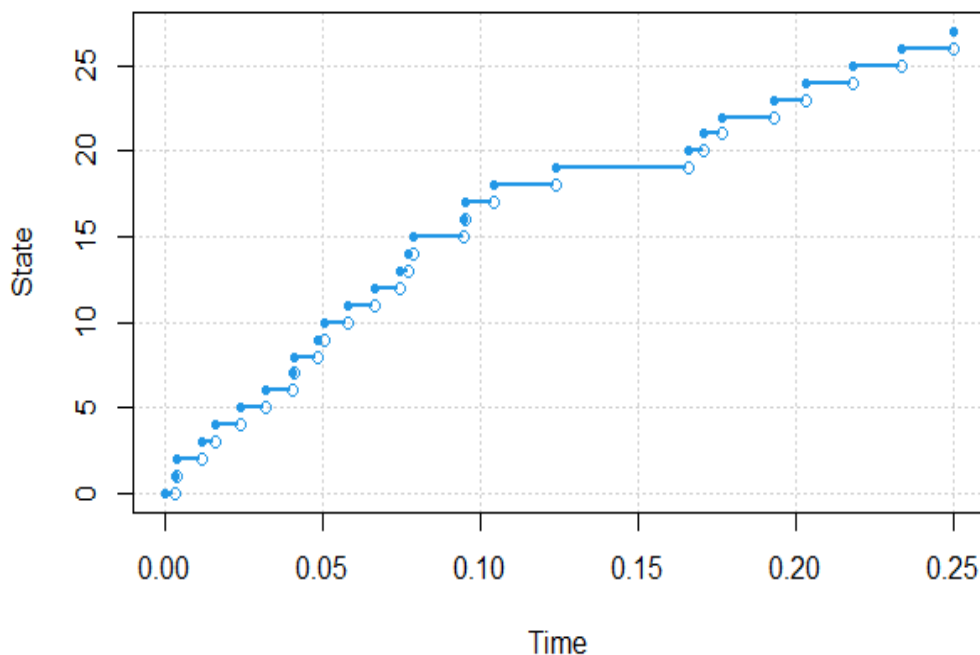
for (i in 2:(njumps+1)) {
  if (runif(1)<= lambda(time.star[i])/lambda.star(time.star[i]))
    accepted[i]=1 else accepted[i]=0
}

time<- time.star[-which(accepted==0)]
N<- 0:(length(time)-1)

#plotting trajectory
plot(time, N, type="n", xlab="Time", ylab="State", panel.first = grid())
segments(time[-length(time)],N[-length(time)], time[-1]-0.001, N[-length(time)],
lwd=2, col=4)

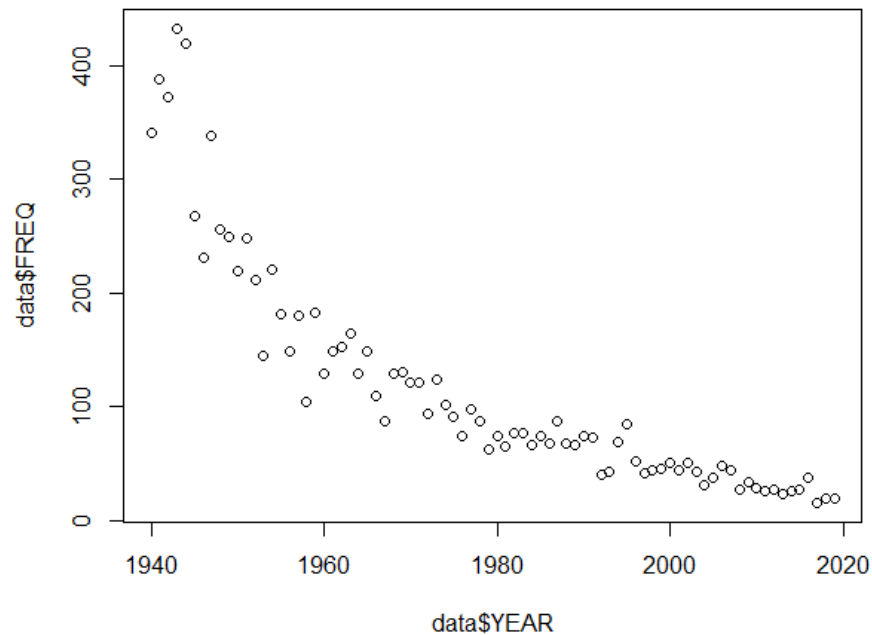
points(time, N, ylim=c(0,120), pch=20, col=4)
points(time[-1],N[-length(time)], pch=1, col=4)

```



EXERCISE 4.6. (a) We run the following code to plot the counts of lightning deaths against year.

```
data<- read.csv(file="./lightningdata.csv", header=TRUE, sep=",")
plot(data$YEAR, data$FREQ)
```



We can see from the plot that the intensity rate decreases over time roughly exponentially. A possible explanation for it is that over the years more awareness has been created among citizens through educational efforts, so fewer people are exposed to the hazard.

(b) We can see that lightning strikes are essentially a seasonal phenomenon. The majority of them happen between May and September. It means that some interarrival times have very large values not inherent to an exponential distribution. Moreover, some incidents resulted in multiple fatalities which would be an event of probability zero under the Poisson law.

EXERCISE 4.7. (a) The code and output below estimate the parameters of the model using the regression approach.

```
port.data<- read.csv(file="./Exercise4.4Data.csv", header=TRUE, sep=",")

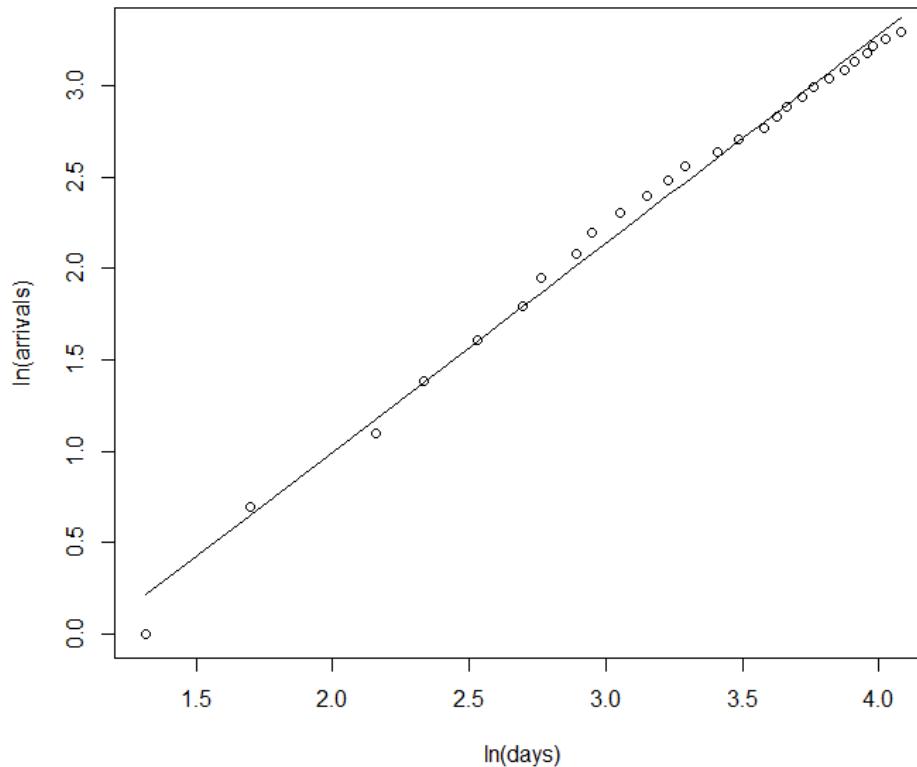
x<- log(port.data$days)
y<- log(port.data$arrivals)

glm(y~x)

plot(x,y, xlab="ln(days)", ylab="ln(arrivals)")

Coefficients:
(Intercept)          x
      -1.294         1.145

lines(x, -1.294+1.145*x)
```



The estimates of the model parameters are $\hat{\alpha} = e^{-1.294} = 0.274172$, and $\hat{\beta} = 1.145$.

(b) The code that follows estimates the parameters using the maximum likelihood approach.

```
port.data<- read.csv(file="./Exercise4.4Data.csv", header=TRUE, sep=",")

x<- log(port.data$days)
y<- log(port.data$arrivals)
N<- 27

print(beta.hat<- N/(N*x[N]-sum(x)))

1.162782

print(alpha.hat<- N/exp(x[N]*beta.hat))

0.2351766
```

The MLEs are $\hat{\alpha} = 0.2351766$, and $\hat{\beta} = 1.162782$.

(c) To predict when the next 10,000 TEUs arrive at the port, we submit the following lines of code.

```

port.data<- read.csv(file="./Exercise4.4Data.csv", header=TRUE, sep=",")

x<- log(port.data$days)
y<- log(port.data$arrivals)
N<- 27

alpha.hat<- c(0.274172, 0.2351766)
beta.hat<- c(1.145, 1.162782)
S.hat<- c()

library(pracma)

for(i in 1:2)
print(S.hat[i]<- alpha.hat[i]^(-
1/beta.hat[i])*exp(alpha.hat[i]*exp(x[N])^beta.hat[i])*
gammainc(alpha.hat[i]*exp(x[N])^beta.hat[i], 1/beta.hat[i]+1)[2])

```

```

60.85581
60.97308

```

According to the data, the 27th arrival was on day 59.1. The 28th arrival is predicted to be on day 60.85581 (by the linear regression), or 60.97308 (by the maximum likelihood).

CHAPTER 5

EXERCISE 5.1. (a) Let $X(t) = \sum_{i=1}^{N(t)} Y_i$ be the total amount paid in prizes up to time t hours. We know that it is a compound Poisson process with $N(t) \sim \text{Poisson}(1.5t)$, and Y_i independent of each other and $N(t)$. The first two moments of Y_1 are $E(Y_1) = (\$5000)(0.15) + (2000)(0.35) + (\$500)(0.2) + (\$100)(0.3) = \$1,580$, and $E(Y_1^2) = (\$5000)^2(0.15) + (2000)^2(0.35) + (\$500)^2(0.2) + (\$100)^2(0.3) = \$25,203,000$.

Therefore, the mean of $X(200)$ is $E(X(200)) = (1.5)(200)(\$1,580) = \$474,000$. The variance is $\text{Var}(X(200)) = (1.5)(200)(\$25,203,000) = \$2,156,900,000$, and the standard deviation is

$$\sqrt{\text{Var}(X(200))} = \sqrt{\$2,156,900,000} = \$39,508.23.$$

The budget for 100 games should be $E(X(200)) + \sqrt{\text{Var}(X(200))} = \$474,000 + \$39,508.23 = \$513,508.23$.

(b) Below are the codes, all relevant output, and the graph for the simulated 100 games.

```
#specifying parameters
lambda<- 1.5
total.hours<- 200
amount<- c(5000, 2000, 500, 100)
p<- c(0.15, 0.35, 0.2, 0.3)

#specifying seed
set.seed(704661)

#generating number of prizes
nprizes<- rpois(1,lambda*total.hours)

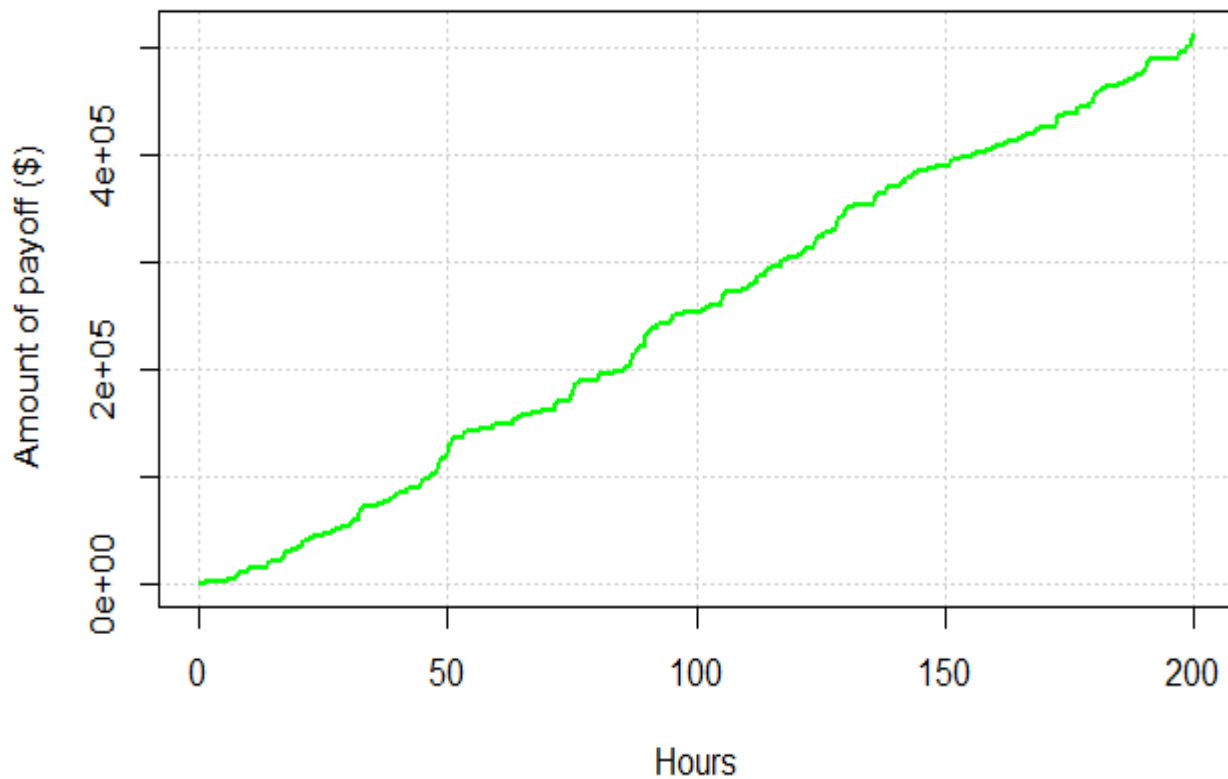
#defining vectors
payoff<- c()
time<- c()
u<- c()

#setting initial values
payoff[1]<- 0
u[1]<- 0

#generating standard uniforms
for(i in 2:(nprizes+1)) {
  u[i]<- runif(1)
  payoff[i]<- payoff[i-1] + amount[sample(1:4, 1, prob=p)]
}

#computing event times
hour<- total.hours*sort(u)

#simulating trajectory
plot(hour, payoff, type="l", lty=1, lwd=2, col="green", xlab="Hours", ylab="Amount
of payoff ($)", panel.first = grid())
```



```
nprizes
```

```
308
```

```
payoff[length(payoff)]
```

```
557100
```

There were a total of 308 prizes given out during the 200 hours of the 100 games. The total payoff was \$557,100.00. Since the budget that the producer had for the 100 games was \$513,508.23, the producer ran out of money before the 100th game.

```
payoff
```

```
[289] 511200 513200 513300 515300 515400
```

The producer ran out of budget after the 291st prize was given out.

```
hour[291]
```

```
190.8272
```

The 291st prize was given out at hour 190.8272 of the show, that is, during the first half of the 96th show.

EXERCISE 5.2. (a) Let $Y_i \sim \text{Unif}(\$30, \$300)$ be the i th claim amount. The two first moments of Y_1 are $E(Y_1) = \frac{\$30 + \$300}{2} = \$165$, and $E(Y_1^2) = \frac{1}{\$300 - \$30} \int_{\$30}^{\$300} u^2 du = \frac{(\$300)^3 - (\$30)^3}{3(\$270)} = \$^2 33,300$.

Let $X(t) = \sum_{i=1}^{N(t)} Y_i$ be the aggregate claim process. Its mean for $t = 30$ is $E(X(30)) = (60)(30)(\$165) = \$297,000$, and the standard deviation is $\sqrt{Var(X(30))} = \sqrt{(60)(30)(\$^2 33,300)} = \$7,742.093$.

(b) By the Central Limit Theorem, $Z = \frac{X(30) - E(X(30))}{\sqrt{Var(X(30))}} = \frac{X(30) - 297000}{7742.093}$ has approximately a $N(0,1)$ distribution, and therefore, $P(X(30) > 300000) = P\left(Z > \frac{300000 - 297000}{7742.093}\right) = P(Z > 0.387492) = 0.349196$.

EXERCISE 5.3. (a) Let $X(t) = \sum_{i=1}^{N(t)} Y_i$ denote the aggregate number of light photons that are generated up to t seconds. We are given that $N(t) \sim \text{Poisson}(\lambda t)$ and $Y_i \sim \text{Poisson}(\tilde{\lambda})$. The mean and standard deviation of $X(t)$ are $E(X(t)) = (\lambda)(t)(\tilde{\lambda})$ and $\sqrt{Var(X(t))} = \sqrt{(\lambda)(t)(EY_1^2)} = \sqrt{(\lambda)(t)(\tilde{\lambda} + \tilde{\lambda}^2)}$.

(b) The code below simulates 100 values of the aggregate number of light photons generated within 10 seconds.

```
#specifying parameters
total.time<- 10
lambda<- 50
lambda.tilde<- 5

total.photons<- c()

for (j in 1:100) {

  nphotons<- c()
  nphotons[1]<- 0

  #generating N(t)
  set.seed(150*j)
  N<- rpois(1,lambda*total.time)

  #simulating trajectory
  for (i in 2:N)
    nphotons[i]<- nphotons[i-1]+rpois(1,lambda.tilde)

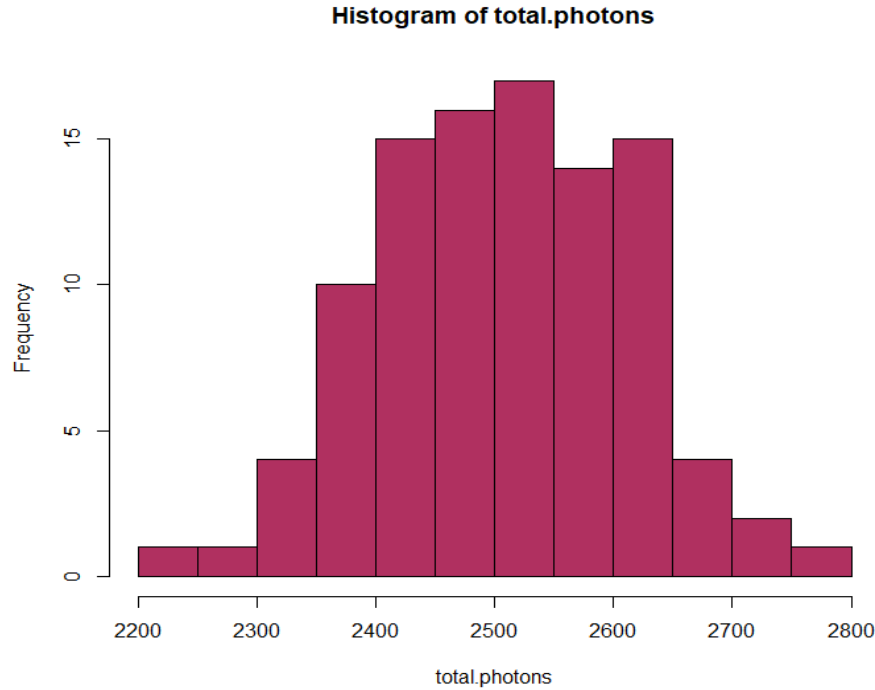
  total.photons[j]<- nphotons[N]
}
```

```
total.photons
[1] 2333 2737 2532 2392 2612 2484 2474 2440
[9] 2638 2549 2549 2361 2459 2603 2633 2537
[17] 2379 2314 2496 2630 2410 2233 2350 2610
[25] 2506 2412 2444 2444 2572 2430 2477 2472
[33] 2538 2470 2648 2641 2333 2425 2475 2549
[41] 2382 2599 2451 2386 2477 2262 2673 2627
[49] 2493 2490 2789 2449 2572 2537 2566 2427
[57] 2591 2469 2414 2508 2610 2587 2523 2384
[65] 2695 2578 2633 2523 2355 2597 2577 2558
[73] 2550 2396 2406 2496 2576 2627 2434 2552
```

```
[81] 2425 2393 2504 2664 2485 2408 2625 2500
[89] 2595 2609 2521 2524 2420 2569 2659 2741
[97] 2360 2617 2546 2505
```

(c) Below we construct a histogram for these 100 values.

```
hist(total.photons, col="maroon")
```



The histogram does resemble a bell shape. It should be the case since λ is large, and so, the Central Limit Theorem should be applicable.

EXERCISE 5.4. The average present value of the total claim amount is computed by conditioning on the value of $N(t)$. We proceed as follows: $E[P(t)] = EE[P(t)|N(t)] = EE\left[\sum_{i=1}^{N(t)} X_i e^{-\delta S_i} \mid N(t)\right] = E\left[\sum_{i=1}^{N(t)} E(X_i)E(e^{-\delta S_i})\right]$. The claim arrival times $S_1, \dots, S_{N(t)}$ are order statistics from a uniform distribution on $[0, t]$ and we can write $S_i = tU_{(i)}$ where $U_{(i)}$ is the i th order statistic from the standard uniform distribution. We continue $E[P(t)] = E\left[\sum_{i=1}^{N(t)} E(X_i)E(e^{-\delta t U_{(i)}})\right] = E(X_1)E\left[\sum_{i=1}^{N(t)} E(e^{-\delta t U_i})\right] = E(X_1)E(N(t))E(e^{-\delta t U_1}) = E(X_1)\lambda \int_0^1 e^{-\delta t u} du = E(X_1)\left(\frac{\lambda}{\delta}\right)(1 - e^{-\delta t})$.

EXERCISE 5.5. (a) Random variables Y_i 's are iid $\sim \text{Poisson}(\beta)$. Therefore, $\sum_{i=1}^n Y_i \sim \text{Poisson}(\beta n)$. We write $P(X(t) = x) = P(\sum_{i=1}^{N(t)} Y_i = x) = \sum_{n=0}^{\infty} P(\sum_{i=1}^n Y_i = x \mid N(t) = n)P(N(t) = n) = \sum_{n=0}^{\infty} \frac{(\beta n)^x}{x!} e^{-\beta n} \frac{(\lambda t)^n}{n!} e^{-\lambda t} = \frac{\beta^x}{x!} e^{-\lambda t} \sum_{n=0}^{\infty} \frac{n^x (\lambda t)^n}{n!} e^{-\beta n}$.

(b) $E(X(t)) = \lambda t E(Y_1) = \lambda t \beta$, $\text{Var}(X(t)) = \lambda t E Y_1^2 = \lambda t (\beta + \beta^2)$.

$$(c) P(X(t) = 0) = \frac{\beta^0}{0!} e^{-\lambda t} \sum_{n=0}^{\infty} \frac{n^0 (\lambda t)^n}{n!} e^{-\beta n} = e^{-\lambda t} \sum_{n=0}^{\infty} \frac{(\lambda t e^{-\beta})^n}{n!} = e^{-\lambda t + \lambda t e^{-\beta}} = e^{-\lambda t(1 - e^{-\beta})}.$$

The ratio between the variance and mean is $\frac{Var(X(t))}{E(X(t))} = 1 + \beta$, thus, β can be estimated as

$$\hat{\beta} = \frac{\hat{Var}(X(t))}{\hat{E}(X(t))} - 1. \text{ Also, } \ln P(X(t) = 0) = \ln P(0) = -\lambda t(1 - e^{-\beta}). \text{ Hence, we can estimate } \lambda \text{ by}$$

$$\hat{\lambda} = -\frac{\ln \hat{P}(0)}{t(1 - e^{-\hat{\beta}})}.$$

EXERCISE 5.6. (a) The total dollar amount can be modeled by a compound Poisson process $X(t) = \sum_{i=1}^{N(t)} Y_i$ where $N(t)$ is the process governing the number of cars that come to the gas station up to time t , and Y_i is the dollar amount that the i th car driver pays. We are given that $N(t) \sim \text{Poisson}(\lambda t)$, and $Y_i \sim \text{Gamma}(\alpha, \beta)$ with mean $E(Y_i) = \alpha\beta$, and variance $Var(Y_i) = \alpha\beta^2$.

(b) Denote by $T_i \sim \text{Exp}(\text{mean} = \frac{1}{\lambda})$ the interarrival times between car arrivals. The method of moments estimator of λ is $\hat{\lambda} = 1/\bar{T}$, the reciprocal of the sample mean of the interarrival times. The method of moments estimators of α and β are $\hat{\alpha} = \frac{n\bar{Y}^2}{\sum_{i=1}^n Y_i^2 - n\bar{Y}^2}$ and $\hat{\beta} = \frac{\bar{Y}}{\hat{\alpha}} = \frac{\sum_{i=1}^n Y_i^2 - n\bar{Y}^2}{n\bar{Y}}$. They solve the system of two equations: $\bar{Y} = \hat{E}(Y_1) = \hat{\alpha}\hat{\beta}$ and $\frac{1}{n}\sum_{i=1}^n Y_i^2 = \hat{E}(Y_1^2) = \hat{Var}(Y_1) + (\hat{E}(Y_1))^2 = \hat{\alpha}\hat{\beta}^2 + (\hat{\alpha}\hat{\beta})^2 = \hat{\beta}\bar{Y} + \bar{Y}^2$.

(a) The code below produces numeric values of the estimators and plots histograms with fitted curves.

```
gas.data<- read.csv(file="./Exercise5.6Data.csv", header=TRUE, sep=",")

#computing lag
gas.data$ArrivalTime.lag<- c(0,head(gas.data$ArrivalTime, -1))
#gas.data<-gas.data[-1,] #removing first row

#computing interarrival times
interarrival.time<- gas.data$ArrivalTime-gas.data$ArrivalTime.lag

#estimating lambda of Poisson arrival
print(lambda.hat<- 1/mean(interarrival.time))

0.6029832

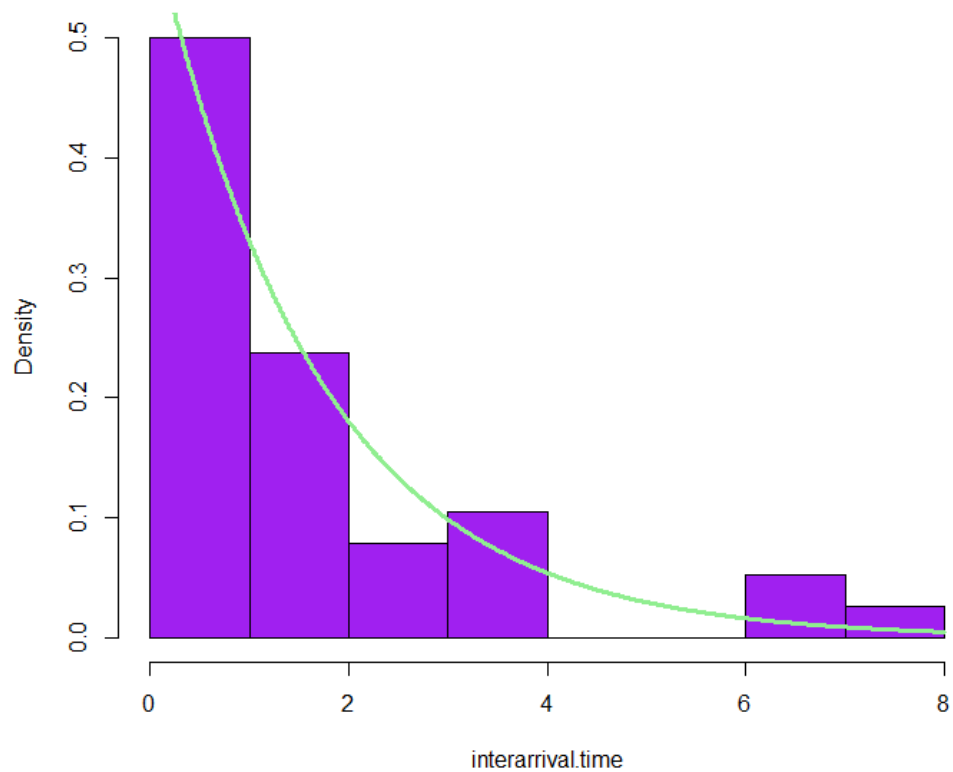
print(1/lambda.hat)

1.658421
```

There are, on average, 0.6029832 car arrivals every minute. The average wait time between two arrivals is 1.658421 minutes.

```
#overlaying histogram and fitted exponential density curve
hist(interarrival.time,freq=FALSE, col="purple")
x<- seq(0, 8, by=0.01)
y<- dexp(x,lambda.hat)
lines(x, y, lty=1, col="light green", lwd=3)
```

Histogram of interarrival.time



```
#estimating parameters of gamma distribution
amount<- gas.data$AmountSpent
print(alpha.hat<- length(amount)*mean(amount)^2/(sum(amount^2)-
length(amount)*mean(amount)^2))
```

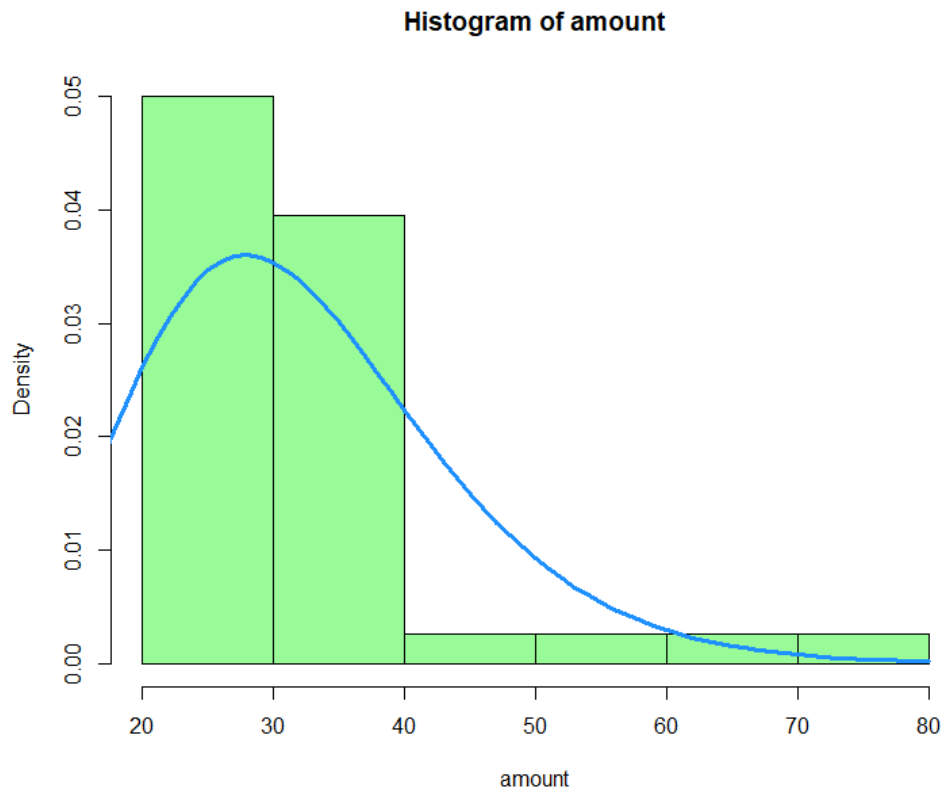
7.48998

```
print(beta.hat<- mean(amount)/alpha.hat)
```

4.29609

```
#overlying histogram and fitted density
hist(amount, freq=FALSE, col = "pale green")
```

```
x<- 0:80
y<- dgamma(x, alpha.hat, 1/beta.hat)
lines(x, y, lty=1, col="dodger blue", lwd=3)
```



(d) The estimated mean of the total dollar amount at one hour is $\hat{E}(X(60)) = (\hat{\lambda})(t)(\hat{\alpha})(\hat{\beta}) = (0.6029832)(60)(7.48998)(4.29609) = \$1,164.154$. The estimated standard deviation is

$$\sqrt{\hat{Var}(X(60))} = \sqrt{(\hat{\lambda})(t)(\hat{\alpha})(\hat{\beta})^2} = \sqrt{(0.6029832)(60)(7.48998)(4.29609)^2} = \$70.71995.$$

CHAPTER 6

EXERCISE 6.1. (a) $Cov(N(s), N(t) - N(s)) = E[N(s)N(t-s)] - E[N(s)]E[N(t-s)] = EE[N(s)N(t-s) | \Lambda] - EE[N(s) | \Lambda] \cdot EE[N(t-s) | \Lambda] = E[(\Lambda s)(\Lambda)(t-s)] - E[\Lambda s]E[\Lambda(t-s)] = s(t-s)E(\Lambda^2) - s(t-s)(E(\Lambda))^2 = s(t-s)Var(\Lambda).$

(b) $Cov(N(s), N(t)) = E[N(s)N(t)] - E[N(s)]E[N(t)] = EE[N(s)(N(t) - N(s) + N(s)) | \Lambda] - E[N(s)]E[N(t)] = EE[N(s)N(t-s) | \Lambda] + EE[(N(s))^2 | \Lambda] - EE[N(s) | \Lambda] \cdot EE[N(t) | \Lambda] = E[(\Lambda s)(\Lambda)(t-s)] + E[\Lambda s + (\Lambda s)^2] - E(\Lambda s)E(\Lambda t) = s(t-s)E(\Lambda^2) + sE(\Lambda) + s^2E(\Lambda^2) - st(E(\Lambda))^2 = stE(\Lambda^2) - st(E(\Lambda))^2 + sE(\Lambda) = stVar(\Lambda) + sE(\Lambda).$

EXERCISE 6.2. (a) $F_{\Lambda|N(t)}(\lambda|n) = P(\Lambda \leq \lambda | N(t) = n) = \frac{P(N(t)=n, \Lambda \leq \lambda)}{P(N(t)=n)}$

$$= \frac{\int_0^\lambda P(N(t) = n | \Lambda = u) f_\Lambda(u) du}{\int_0^\infty P(N(t) = n | \Lambda = u) f_\Lambda(u) du} = \frac{\int_0^\lambda \frac{(ut)^n}{n!} e^{-ut} f_\Lambda(u) du}{\int_0^\infty \frac{(ut)^n}{n!} e^{-ut} f_\Lambda(u) du} = \frac{\int_0^\lambda u^n e^{-ut} f_\Lambda(u) du}{\int_0^\infty u^n e^{-ut} f_\Lambda(u) du}.$$

$$(b) f_{\Lambda|N(t)}(\lambda|n) = F'_{\Lambda|N(t)}(\lambda|n) = \frac{\lambda^n e^{-\lambda t} f_\Lambda(\lambda)}{\int_0^\infty \lambda^n e^{-\lambda t} f_\Lambda(\lambda) d\lambda}.$$

$$(c) E[\Lambda | N(t) = n] = \int_0^\infty \lambda f_{\Lambda|N(t)}(\lambda|n) d\lambda = \frac{\int_0^\infty \lambda^{n+1} e^{-\lambda t} f_\Lambda(\lambda) d\lambda}{\int_0^\infty \lambda^n e^{-\lambda t} f_\Lambda(\lambda) d\lambda}.$$

EXERCISE 6.3. (a) Denote by $\{N(t), t \geq 0\}$ the process of visitor arrival. We know that $N(t) \sim \text{Poisson}(\Lambda t)$ where $P(\Lambda = 4) = 0.46, P(\Lambda = 2) = 0.24$, and $P(\Lambda = 3) = 0.30$. The mean and variance of $N(t)$ are $E(N(t)) = tE(\Lambda) = t((4)(0.46) + (2)(0.24) + (3)(0.30)) = 3.22t$, and $Var(N(t)) = t^2Var(\Lambda) + tE(\Lambda) = t^2((4)^2(0.46) + (2)^2(0.24) + (3)^2(0.30) - (3.22)^2) + 3.22t = 0.6516t^2 + 3.22t$.

(b) The code below simulates 5 trajectories of the process with 200 visitors each.

```
#specifying parameters
p<- c(0.46, 0.24, 0.30)
lambda<- c(4, 2, 3)
nvisitors<- 200
time<- data.frame()
N<- data.frame()

#specifying seed
set.seed(109088)

#creating loop to simulate trajectories
for(j in 1:5) {

#selecting rate
Lambda<- lambda[sample(1:3, 1, prob=p)]
```

```

#setting initial values
time[1,j]<- 0
N[1,j]<- 0

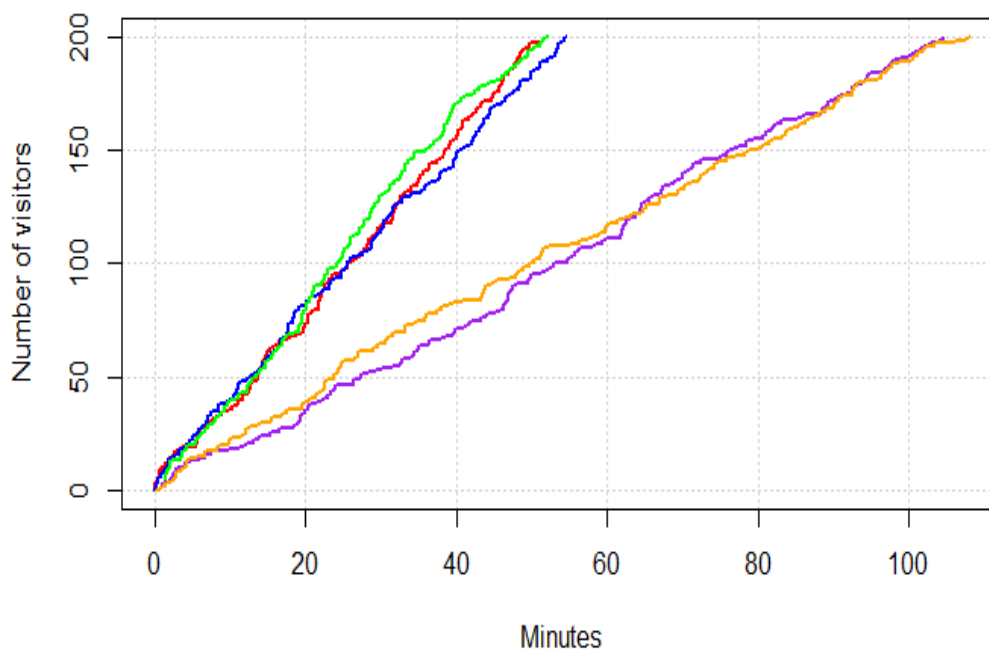
#simulating trajectory
i<- 2

repeat {
time[i,j]<- time[i-1,j]+round((-1/Lambda)*log(1-runif(1)),3)-0.001
  N[i,j]<- N[i-1,j]

if(i==2*nvisitors+2) break
else {
  time[i+1,j]<- time[i,j]+0.001
  N[i+1,j]<- N[i,j]+1
  i<- i+2
}
}

#plotting trajectories
matplot(time, N, type="l", lty=1, lwd=2, col=c("red", "blue", "green",
"purple", "orange"), xlab="Minutes", ylab="Number of visitors",
panel.first=grid())

```



(c) The code below simulates 5 trajectories of the process that depict arrivals within one hour.

```

#specifying parameters
t<- 60
p<- c(0.46, 0.24, 0.30)
lambda<- c(4, 2, 3)
time<- data.frame()
N<- data.frame()

#specifying seed

```

```

set.seed(5055562)

#creating loop to simulate trajectories
for(j in 1:5) {

#selecting rate
Lambda<- lambda[sample(1:3, 1, prob=p)]

#setting initial values
time[1,j]<- 0
N[1,j]<- 0

#generating N(t)
N.total<- rpois(1,Lambda*t)

#generating N(t) standard uniforms
u<- 1:N.total
for(i in 1:N.total)
  u[i]<- runif(1)

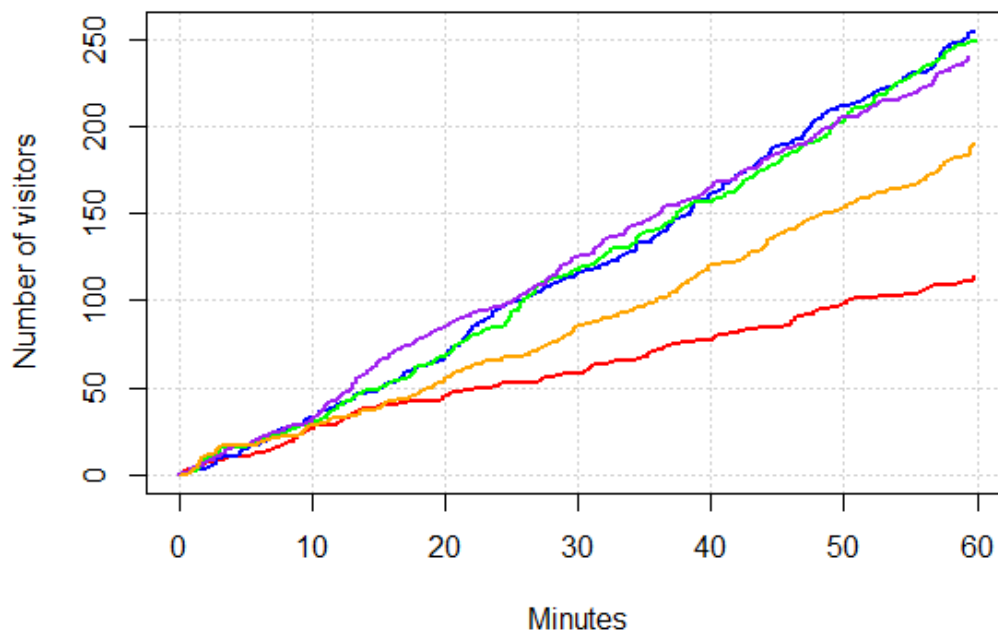
#sorting standard uniforms
u.sorted<- sort(u)

#computing N(t) event times
s<- t*u.sorted

#generating jumps
for (i in seq(2, 2*N.total, 2)) {
  time[i,j]<- s[i/2]-0.001
  time[i+1,j]<- s[i/2]
  N[i,j]<- N[i-1,j]
  N[i+1,j]<- N[i-1,j]+1
}
}

#plotting simulated trajectories
matplot(time, N, type="l", lty=1, lwd=2, col=c("red", "blue", "green",
"purple", "orange"), xlab="Minutes", ylab="Number of visitors",
panel.first=grid())

```



EXERCISE 6.4. (a) Let $N(t)$ denote the number of defaults by time t . It is given that $N(t) \sim \text{Poisson}(\Lambda t)$ where $\Lambda \sim \text{Uniform}(0, 2)$. The average number of defaults within 5 years is $E(N(5)) = tE(\Lambda) = (5)(1) = 5$.

(b) The variance of the number of defaults within 5 years is $\text{Var}(N(5)) = t^2 \text{Var}(\Lambda) + tE(\Lambda) = (5)^2 \left(\frac{2^2}{12}\right) + (5)(1) = 13.3333$.

(c) As shown in Exercise 1(a), $\text{Cov}(N(s), N(t) - N(s)) = s(t - s)\text{Var}(\Lambda)$. For $s = 3$, and $t = 5$, $\text{Cov}(N(3), N(5) - N(3)) = (3)(5 - 3)(1/3) = 2$.

(d) As shown in Exercise 1(b), $\text{Cov}(N(s), N(t)) = st\text{Var}(\Lambda) + sE(\Lambda)$. For $s = 3$, and $t = 5$, $\text{Cov}(N(3), N(5)) = (3)(5)(1/3) + (3)(1) = 8$.

(e) Using the result proven in Exercise 6.2, we get $P(\Lambda < 0.5 \mid N(5) = 2) = \frac{\int_0^{0.5} u^2 e^{-5u} f_{\Lambda}(u) du}{\int_0^{\infty} u^2 e^{-5u} f_{\Lambda}(u) du} = \frac{\int_0^{0.5} u^2 e^{-5u} du}{\int_0^{\infty} u^2 e^{-5u} du}$. Now, $\int u^2 e^{-5u} du = -\frac{1}{5}u^2 e^{-5u} - \frac{2}{25}ue^{-5u} - \frac{2}{125}e^{-5u}$. Therefore,

$$P(\Lambda < 0.5 \mid N(5) = 2) = \frac{\int_0^{0.5} u^2 e^{-5u} du}{\int_0^{\infty} u^2 e^{-5u} du} = \frac{-\frac{1}{5}u^2 e^{-5u} - \frac{2}{25}ue^{-5u} - \frac{2}{125}e^{-5u} \Big|_0^{0.5}}{-\frac{1}{5}u^2 e^{-5u} - \frac{2}{25}ue^{-5u} - \frac{2}{125}e^{-5u} \Big|_0^{\infty}} = \frac{-e^{-2.5}(\frac{1}{20} + \frac{1}{25} + \frac{2}{125}) + \frac{2}{125}}{\frac{2}{125}} = 0.456187.$$

EXERCISE 6.5. (a) Let $N(t)$ denote the amount of SWE accumulated within time t . It is given that $N(t) \sim \text{Poisson}(\Lambda t)$ where $\Lambda \sim \text{Poisson}(\lambda = 24.3)$. The average and standard deviation of SWE for one year are $E(N(1)) = tE(\Lambda) = \lambda t = (24.3)(1) = 24.3$ inches, and $\sqrt{\text{Var}(N(1))} = \sqrt{t^2 \text{Var}(\Lambda) + tE(\Lambda)} = \sqrt{t^2 \lambda + t\lambda} = \sqrt{(1^2 + 1)(24.3)} = \sqrt{48.6} = 6.97$ inches. For five years, $E(N(5)) = tE(\Lambda) = \lambda t = (24.3)(5) = 121.5$ inches and $\sqrt{\text{Var}(N(5))} = \sqrt{(5^2 + 5)(24.3)} = \sqrt{729} = 27$ inches.

(b) The code below simulates 5 trajectories that reach 140 inches of SWE each.

```
#specifying parameters
lambda<- 24.3
SWE.inches<- 140
time<- data.frame()
N<- data.frame()

#specifying seed
set.seed(9000004)

#creating loop to simulate trajectories
for(j in 1:5) {

#selecting rate
Lambda<- rpois(1,lambda)
```

```

#setting initial values
time[1,j]<- 0
N[1,j]<- 0

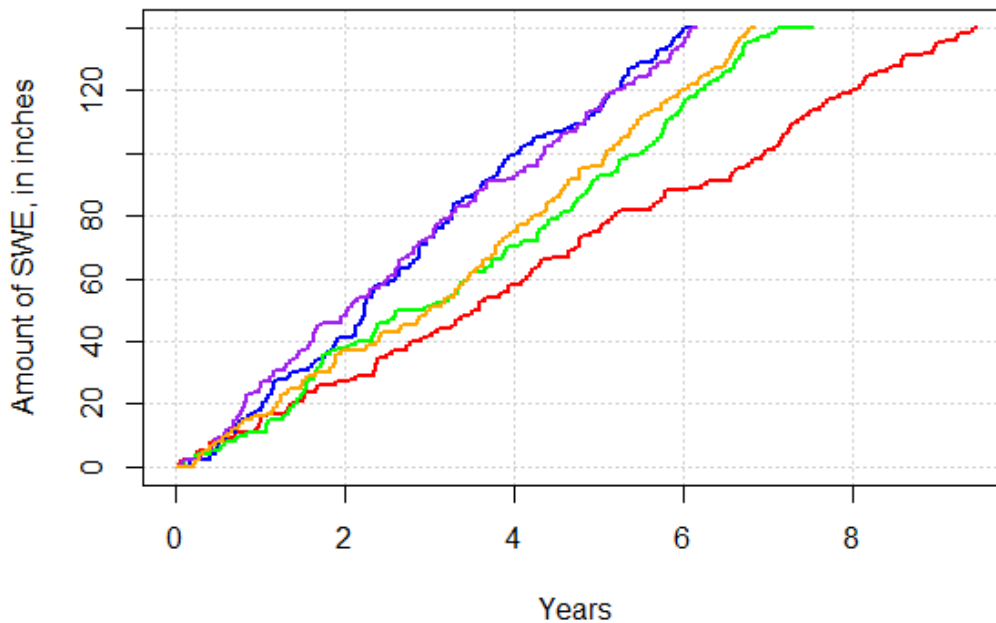
#simulating trajectory
i<- 2

repeat {
time[i,j]<- time[i-1,j]+round((-1/Lambda)*log(1-runif(1)),3)-0.001
  N[i,j]<- N[i-1,j]

if(i==2*SWE.inches+2) break
else {
  time[i+1,j]<- time[i,j]+0.001
  N[i+1,j]<- N[i,j]+1
  i<- i+2
}
}

#plotting trajectories
matplot(time, N, type="l", lty=1, lwd=2, col=c("red", "blue", "green",
"purple", "orange"), xlab="Years", ylab="Amount of SWE, in inches",
panel.first=grid())

```



(c) The code below simulates 5 trajectories spanning over 7 years.

```

#specifying parameters
t<- 7
lambda<- 24.3
time<- data.frame()
N<- data.frame()

#specifying seed
set.seed(1001117)

```



```

#creating loop to simulate trajectories
for(j in 1:5) {

#selecting rate
Lambda<- rpois(1,lambda)

#setting initial values
time[1,j]<- 0
N[1,j]<- 0

#generating N(t)
N.total<- rpois(1,Lambda*t)

#generating N(t) standard uniforms
u<- 1:N.total
for(i in 1:N.total)
  u[i]<- runif(1)

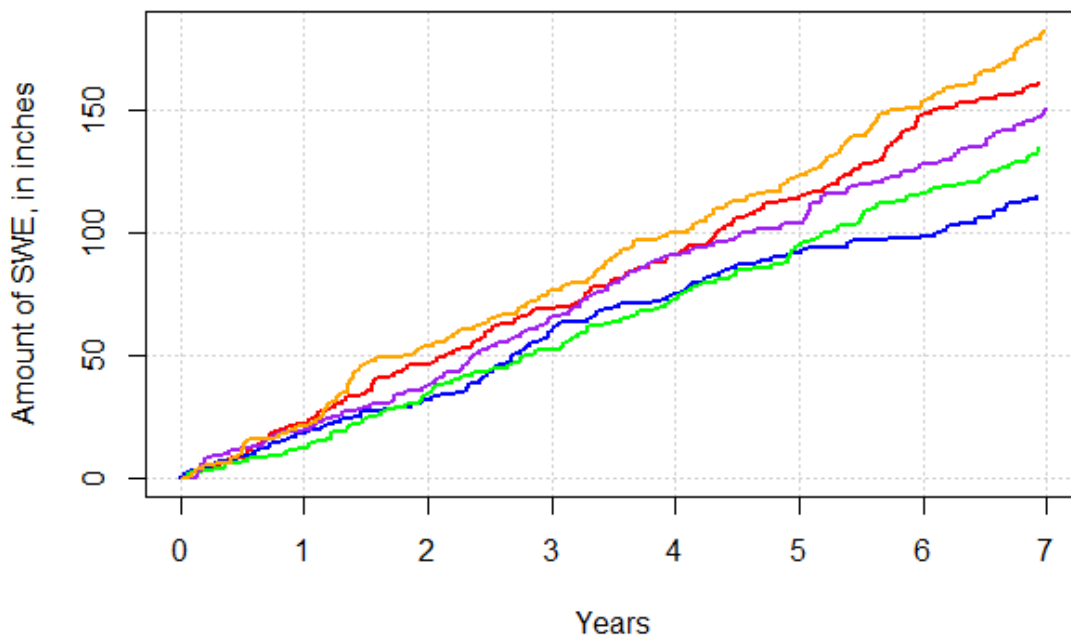
#sorting standard uniforms
u.sorted<- sort(u)

#computing N(t) event times
s<- t*u.sorted

#generating jumps
for (i in seq(2, 2*N.total, 2)) {
  time[i,j]<- s[i/2]-0.001
  time[i+1,j]<- s[i/2]
  N[i,j]<- N[i-1,j]
  N[i+1,j]<- N[i-1,j]+1
}
}

#plotting simulated trajectories
matplot(time, N, type="l", lty=1, lwd=2, col=c("red", "blue", "green",
"purple", "orange"), xlab="Years", ylab="Amount of SWE, in inches",
panel.first=grid())

```



EXERCISE 6.6. (a) Let $N(\ell)$ be the number of defects in an ℓ -yard roll of fabric. We know that $N(\ell) \sim \text{Poisson}(\Lambda\ell)$ where $\Lambda \sim \text{Gamma}(\alpha, \beta)$ with $E(\Lambda) = \alpha/\beta = 0.07$ and $\sqrt{\text{Var}(\Lambda)} = \sqrt{\alpha/\beta^2} = 0.01$. The mean and standard deviation of $N(40)$ are $E(N(40)) = \ell E(\Lambda) = (40)(0.07) = 2.8$ and $\sqrt{\text{Var}(N(40))} = \sqrt{\ell^2 \text{Var}(\Lambda) + \ell E(\Lambda)} = \sqrt{(40)^2(0.01)^2 + (40)(0.07)} = 1.72$.

(b) As derived in Application 6.2(b), for a given $N(t) = n$, the conditional distribution of Λ is gamma with parameters $n + \alpha$ and $\ell + \beta$. The parameters α and β solve $\begin{cases} \alpha/\beta = 0.07 \\ \alpha/\beta^2 = 0.0001 \end{cases}$. From here, $\alpha = 49$ and $\beta = 700$. We are also given that $\ell = 40$ and $n = 4$.

The line of code given below computes the probability to be above 0.08 for a gamma distribution with parameters $n + \alpha = 4 + 49 = 53$ and $\ell + \beta = 40 + 700 = 740$.

```
pgamma(0.08, 53, 740, lower.tail=FALSE)
```

```
0.1932722
```

CHAPTER 7

EXERCISE 7.1. (a) We plug $\lambda_n = n\lambda$ and $\mu_n = 0$ into (7.1), and note that n starts with 1 and not 0. The Kolmogorov forward equations become $P'_1(t) = -\lambda P_1(t)$ and $P'_n(t) = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t)$, $n = 2, 3, \dots$, with the initial condition $P_1(0) = 1$.

(b) To show that $P_n(t) = e^{-\lambda t}(1 - e^{-\lambda t})^{n-1}$, $n = 1, 2, \dots$, solve the Kolmogorov equations, we write $P_1(t) = e^{-\lambda t}$, so $P'_1(t) = -\lambda e^{-\lambda t} = -\lambda P_1(t)$. Also,

$$P'_n(t) = -\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-1} + e^{-\lambda t}(n-1)\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-2} = -\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-1} + e^{-\lambda t}(n-1)\lambda(e^{-\lambda t} - 1 + 1)(1 - e^{-\lambda t})^{n-2} = -\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-1} - (n-1)\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-1} + (n-1)\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-2} = (n-1)\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-2} - n\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-1} = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t).$$

(c) The distribution of $X(t)$ is geometric that models the number of trials until the first success where the probability of success is $p = e^{-\lambda t}$. Therefore, $E(X(t)) = \frac{1}{p} = e^{\lambda t}$, and $Var(X(t)) = \frac{1-p}{p^2} = \frac{1-e^{-\lambda t}}{e^{-2\lambda t}} = e^{\lambda t}(e^{\lambda t} - 1)$.

(d) If $\lambda = 4$, the probability that there will be between 3 and 5 particles at week 1 is $P_3(1) + P_4(1) + P_5(1) = e^{-4}(1 - e^{-4})^{3-1} + e^{-4}(1 - e^{-4})^{4-1} + e^{-4}(1 - e^{-4})^{5-1} = 0.051989$. The mean at week 1 is $E(X(1)) = e^4 = 54.59815$, and the standard deviation is $\sqrt{Var(X(1))} = \sqrt{e^4(e^4 - 1)} = 54.09584$.

EXERCISE 7.2. (a) We plug $\lambda_n = n\lambda$ and $\mu_n = 0$ into (7.1) and note that n starts with m and not 0. The Kolmogorov forward equations become $P'_m(t) = -m\lambda P_m(t)$ and $P'_n(t) = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t)$, $n = 2, 3, \dots$, with the initial condition $P_m(0) = 1$.

(b) To verify that $P_n(t) = \binom{n-1}{n-m} e^{-m\lambda t}(1 - e^{-\lambda t})^{n-m}$, $n = m, m+1, \dots$, solve the Kolmogorov equations, we write $P_m(t) = e^{-m\lambda t}$, so $P'_m(t) = -m\lambda e^{-m\lambda t} = -m\lambda P_m(t)$. Further, $P'_n(t) = -m\lambda \binom{n-1}{n-m} e^{-m\lambda t}(1 - e^{-\lambda t})^{n-m} + \binom{n-1}{n-m} e^{-m\lambda t}(n-m)\lambda e^{-\lambda t}(1 - e^{-\lambda t})^{n-m-1} = -m\lambda P_n(t) + \binom{n-1}{n-m} e^{-m\lambda t}(n-m)\lambda(e^{-\lambda t} - 1 + 1)(1 - e^{-\lambda t})^{n-m-1} = -m\lambda P_n(t) - (n-m)\lambda P_n(t) + (n-m)\binom{n-1}{n-m} \lambda e^{-m\lambda t}(1 - e^{-\lambda t})^{n-m-1} = -n\lambda P_n(t) + (n-1)\lambda \binom{n-2}{n-m-1} e^{-m\lambda t}(1 - e^{-\lambda t})^{n-m-1} = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t)$.

(c) The distribution of $X(t)$ is a negative binomial that models the number of trials until the m th success, where the probability of success is $p = e^{-\lambda t}$. Therefore, the mean and the variance are $E(X(t)) = \frac{m}{p} = m e^{\lambda t}$, and $Var(X(t)) = \frac{m(1-p)}{p^2} = m e^{\lambda t}(e^{\lambda t} - 1)$.

(d) $P_{12}(2) = \binom{12-1}{12-5} e^{-(5)(0.2)(2)}(1 - e^{-(0.2)(2)})^{12-5} = 0.0189$. The mean and standard deviations are $E(X(2)) = (5)e^{(0.2)(2)} = 7.459123$, and $\sqrt{Var(X(2))} = \sqrt{(5)e^{0.4}(e^{0.4} - 1)} = 1.915354$.

EXERCISE 7.3. (a) In the Kolmogorov forward equations (7.1), we use $\lambda_n = 0$, and $\mu_n = n\mu$, and the fact that the initial population size is N . We write $P'_N(t) = -N\mu P_N(t)$ and $P'_n(t) = (n+1)\mu P_{n+1}(t) - n\mu P_n(t)$, $n = 0, 1, \dots, N-1$, with the initial condition $P_N(0) = 1$.

(b) The probabilities $P_n(t) = \binom{N}{n} e^{-n\mu t} (1 - e^{-\mu t})^{N-n}$, $n = 0, \dots, N$, solve the Kolmogorov forward equations since $P_N(t) = e^{-N\mu t}$ and so, $P'_N(t) = -N\mu e^{-N\mu t} = -N\mu P_N(t)$. Also,

$$\begin{aligned} P'_n(t) &= -n\mu \binom{N}{n} e^{-n\mu t} (1 - e^{-\mu t})^{N-n} + \binom{N}{n} e^{-n\mu t} (N-n)\mu e^{-\mu t} (1 - e^{-\mu t})^{N-n-1} \\ &= -n\mu P_n(t) + (n+1)\mu \binom{N}{n+1} e^{-(n+1)\mu t} (1 - e^{-\mu t})^{N-(n+1)} = (n+1)\mu P_{n+1}(t) - n\mu P_n(t). \end{aligned}$$

(c) The distribution of $X(t)$ is binomial with parameters N and $p = e^{-\mu t}$. Therefore, $E(X(t)) = Np = Ne^{-\mu t}$, and $Var(X(t)) = Np(1-p) = Ne^{-\mu t}(1 - e^{-\mu t})$.

(d) $P_{12}(3) = \binom{15}{12} e^{-(12)(0.02)(3)} (1 - e^{-(0.02)(3)})^{15-12} = 0.0437$. The mean and standard deviation are $E(X(3)) = 15 e^{-(0.02)(3)} = 14.12647$, and $\sqrt{Var(X(3))} = \sqrt{15 e^{-(0.02)(3)} (1 - e^{-(0.02)(3)})} = 0.907007$.

EXERCISE 7.4. (a) We are given that $\lambda = 1.3$ and $\mu = 0.2$. We need to compute

$$P_4(2) = (1 - P_0) \left(1 - \frac{\lambda}{\mu} P_0\right) \left(\frac{\lambda}{\mu} P_0\right)^{n-1} = (1 - P_0) \left(1 - \frac{1.3}{0.2} P_0\right) \left(\frac{1.3}{0.2} P_0\right)^{4-1}$$

where

$$P_0 = \frac{\mu e^{(\lambda-\mu)t} - \mu}{\lambda e^{(\lambda-\mu)t} - \mu} = \frac{0.2 e^{(1.3-0.2)(2)} - 0.2}{1.3 e^{(1.3-0.2)(2)} - 0.2} = 0.139172.$$

Thus, $P_4(2) = 0.060783$. The mean and variance are $E(X(2)) = e^{(\lambda-\mu)t} = e^{(1.3-0.2)(2)} = 9.025013$ and $Var(X(2)) = \frac{\lambda+\mu}{\lambda-\mu} e^{(\lambda-\mu)t} (e^{(\lambda-\mu)t} - 1) = \frac{1.3+0.2}{1.3-0.2} e^{(1.3-0.2)(2)} (e^{(1.3-0.2)(2)} - 1) = 98.76253$.

(b) Below we simulate a 50-step trajectory of the process that starts in state 1 and has parameters $\lambda = 1.3$ and $\mu = 0.2$.

```
#specifying parameters
lambda<- 1.3
mu<- 0.2
njumps<- 50

#defining state and time as vectors
N<- c()
time<- c()

#setting initial values
N[1]<- 1
time[1]<- 0

#specifying seed
set.seed(353332)
```

```

#simulating trajectory
i<- 2

repeat {
  time.birth<- (-1/(N[i-1]*lambda))*log(1-runif(1))
  time.death<- (-1/(N[i-1]*mu))*log(1-runif(1))

  if(time.birth < time.death | N[i-1]==0) {
    time[i]<- time[i-1] + time.birth - 0.001
    N[i]<- N[i-1]

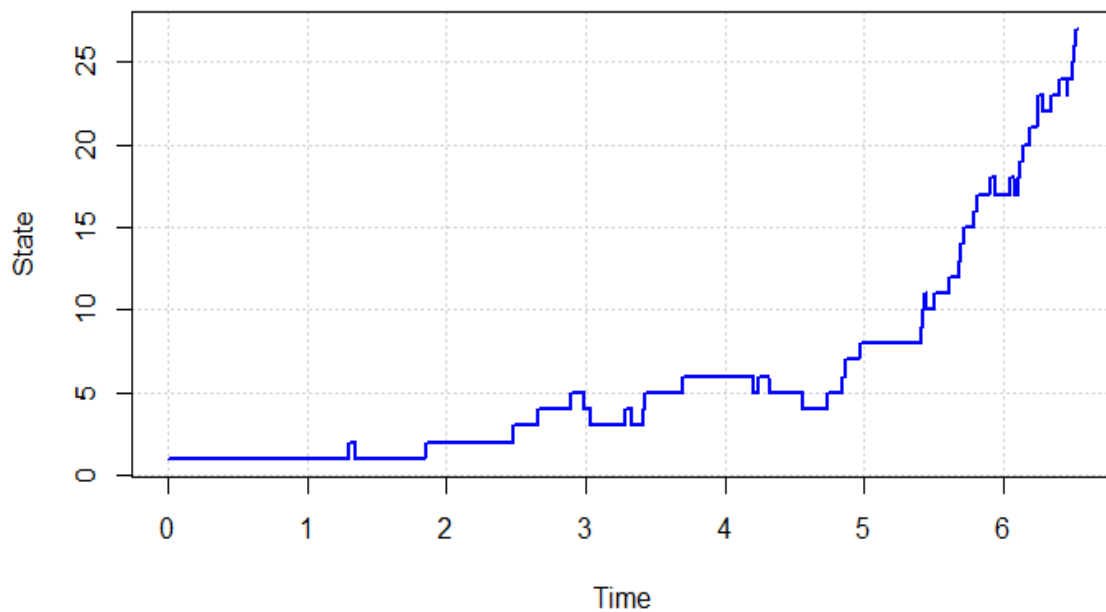
    if(i==2*njumps+2) break
  } else {
    time[i+1]<- time[i] + 0.001
    N[i+1]<- N[i] + 1
    i<- i+2
  }

  if(time.death < time.birth & N[i-1]!=0) {
    time[i]<- time[i-1] + time.death - 0.001
    N[i]<- N[i-1]

    if(i==2*njumps+2) break
  } else {
    time[i+1]<- time[i] + 0.001
    N[i+1]<- N[i] - 1
    i<- i+2
  }
}

#plotting trajectory
plot(time, N, type="l", lty=1, lwd=2, col="blue", xlab="Time", ylab="State",
panel.first=grid())

```



EXERCISE 7.5. (a) If $\lambda > \mu$, the queue will accumulate faster than customers go through the server, and so we expect an infinite number of customers in the system in the long run.

(b) For $\lambda = 3$ and $\mu = 5$, the long-run probability that there will be more than 2 customers in the system is $P(\# \text{ of customers} > 2) = 1 - P_0 - P_1 - P_2 = 1 - \left(1 - \frac{\lambda}{\mu}\right) \mu \left[1 + \frac{\lambda}{\mu} + \left(\frac{\lambda}{\mu}\right)^2\right] = 1 -$

$$\left(1 - \frac{\lambda}{\mu}\right) \frac{1 - \left(\frac{\lambda}{\mu}\right)^3}{1 - \frac{\lambda}{\mu}} = \left(\frac{\lambda}{\mu}\right)^3 = \left(\frac{3}{5}\right)^3 = 0.216.$$

(c) In the long run, the average number of customers in the system is

$$\lim_{t \rightarrow \infty} E(X(t)) = \frac{\lambda}{\mu - \lambda} = \frac{3}{5 - 3} = 1.5.$$

(d) In the long run, the proportion of customers in the system who have to wait more than 1 minute is $P(T > 1) = e^{-(\mu - \lambda)t} = e^{-(5 - 3)(1)} = 0.135335$, or roughly 13.5%.

EXERCISE 7.6. (a) Below we simulate a trajectory of a birth-and-death process with immigration and emigration, with parameters $\lambda = 1, \mu = 0.2, \alpha = 0.3$, and $\beta = 0.1$. The trajectory starts in state 10 and ends in state 25.

```
#specifying parameters
lambda<- 1
mu<- 0.2
alpha<- 0.3
beta<- 0.1

#defining state and time as vectors
N<- c()
time<- c()

#setting initial values
N[1]<- 10
time[1]<- 0

#specifying seed
set.seed(93743765)

#simulating trajectory
i<- 2

repeat {

  time.birth<- (-1/(N[i-1]*lambda+alpha))*log(1-runif(1))
  time.death<- (-1/(N[i-1]*mu+beta))*log(1-runif(1))

  if(time.birth < time.death | N[i-1]==0) {
    time[i]<- time[i-1] + time.birth - 0.001
    N[i]<- N[i-1]

    if(N[i]==25) break
  } else {

    time[i+1]<- time[i] + 0.001
    N[i+1]<- N[i] + 1
  }
}
```

```

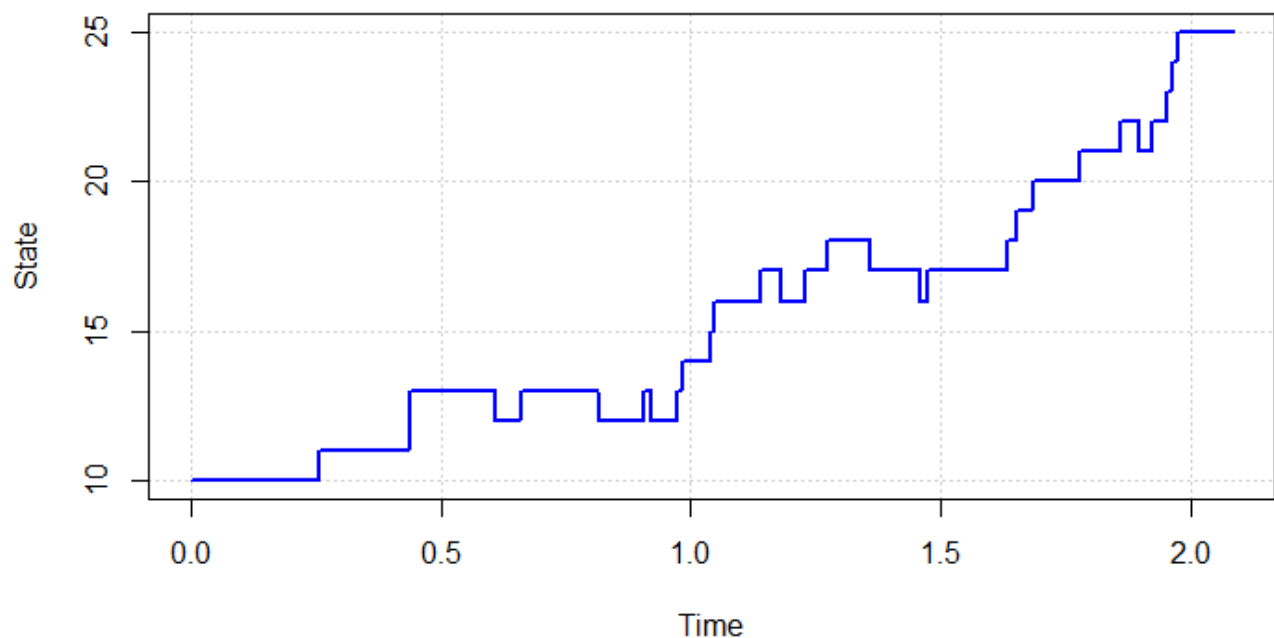
i<- i+2
}
}

if(time.death < time.birth & N[i-1]!=0) {
  time[i]<- time[i-1] + time.death - 0.001
  N[i]<- N[i-1]

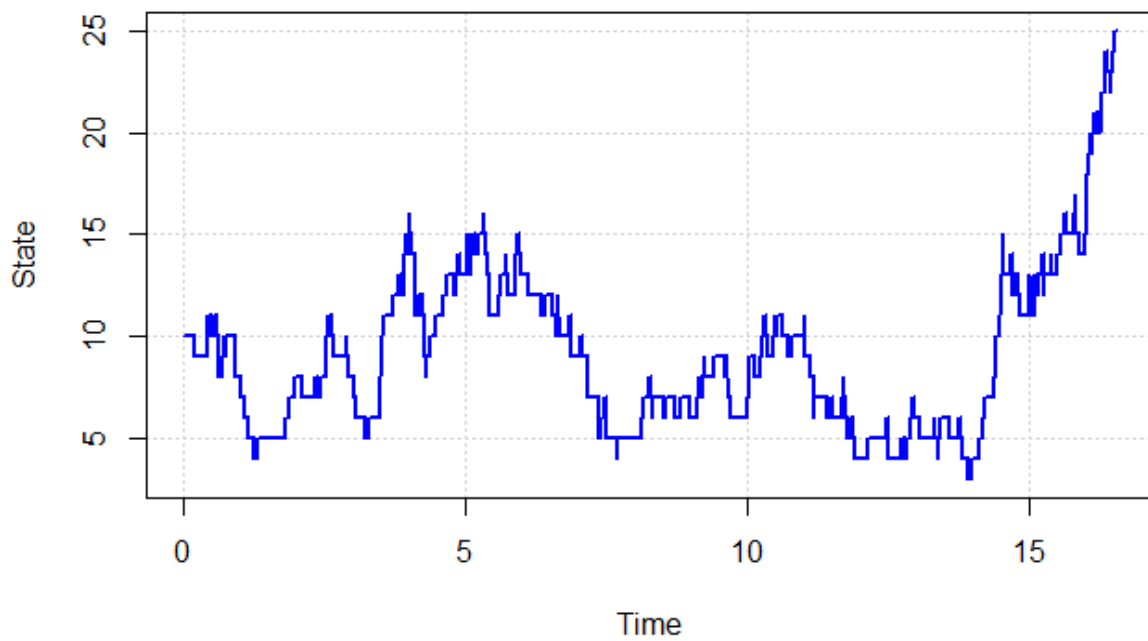
  if(N[i]==25) break
  else {

    time[i+1]<- time[i] + 0.001
    N[i+1]<- N[i] - 1
    i<- i+2
  }
}
}
#plotting trajectory
plot(time, N, type="l", lty=1, lwd=2, col="blue", xlab="Time", ylab="State",
panel.first=grid())

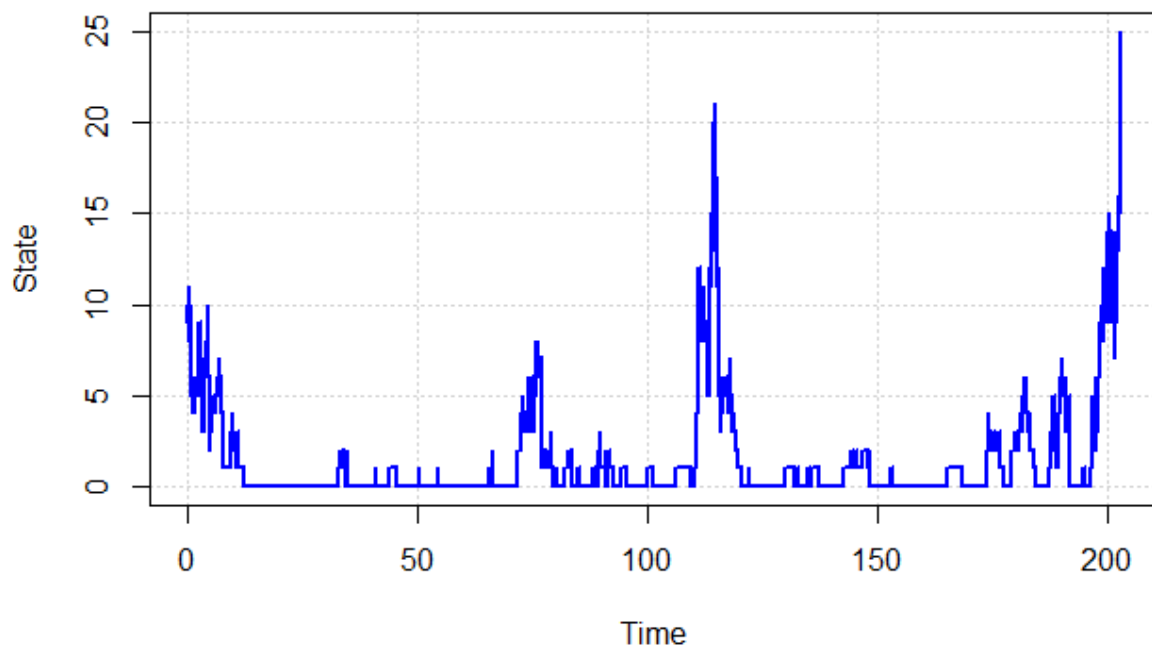
```



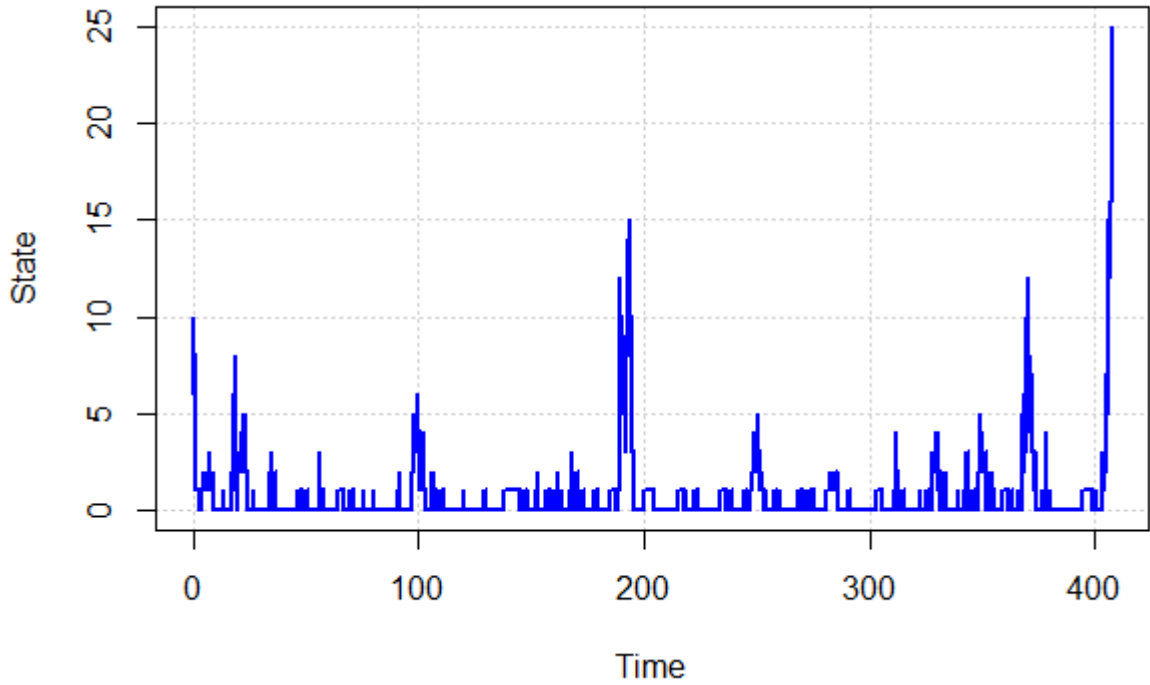
(b) To simulate a trajectory with $\mu = 0.8$, we run the same code but specify the parameter as:
`mu<- 0.8`. The plot is



(c) In the same code we specify `mu<- 1`. The simulated trajectory is



(d) Now we specify `mu<- 1.2`. The graph is



(e) In the plots above we see that as the death rate increases, the population relies on immigration more and more. When $\mu = 0.2$, the flock size grows from 10 to 25 birds within roughly 2 time units. When $\mu = 0.8$, it takes about 16 time units to grow. When $\mu = 1$, it takes about 200 time units to grow. When $\mu = 1.2$, it takes about 400 time units to grow. In the last two cases ($\mu = 1$ and $\mu = 1.2$), the flock keeps dying out and revives due to bird immigration.

CHAPTER 8

EXERCISE 8.1. (a) For each bacterium, the offspring size Z has mean $\mu = E(Z) = (0)(0.25) + (1)(0.15) + (2)(0.6) = 1.35 > 1$, thus the colony growth is a supercritical branching process. The variance of Z is $\sigma^2 = Var(Z) = (0)^2(0.25) + (1)^2(0.15) + (2)^2(0.6) - (1.35)^2 = 0.7275$.

The expected size of the n th generation is $100E(X_n) = 100\mu^n = (100)(1.35)^n$. The variance is $100 Var(X_n) = 100\sigma^2\mu^{n-1} \frac{1-\mu^n}{1-\mu} = (100)(0.7275)(1.35)^{n-1} \frac{(1.35)^n - 1}{1.35 - 1} = (207.8571)(1.35)^{n-1}((1.35)^n - 1)$.

(b) Let $\pi_0 < 1$ denote the extinction probability for descendants of one bacterium. It solves the equation $\pi_0 = \pi_0^0(0.25) + \pi_0^1(0.15) + \pi_0^2(0.6)$, or, equivalently, $0.6\pi_0^2 - 0.85\pi_0 + 0.25 = (\pi_0 - 1)(0.6\pi_0 - 0.25) = 0$. Thus, $\pi_0 = \frac{0.25}{0.6} = 0.4167$.

(c) $P(\text{extinction of descendants of at least one of ten bacteria}) = 1 - P(\text{no extinction}) = 1 - (1 - 0.4167)^{10} = 0.9954$.

EXERCISE 8.2. (a) Let Z denote the size of offspring. We are given that $Z \sim Poi(\lambda)$. The mean of Z is $\mu = E(Z) = \lambda$. If $\lambda > 1$, the process is supercritical; if $\lambda = 1$, the process is critical; if $\lambda < 1$, the process is subcritical.

(b) The variance of the offspring size is $\sigma^2 = Var(Z) = \lambda$. The mean of the size of the n th generation X_n is $E(X_n) = \mu^n = \lambda^n$. The variance is $Var(X_n) = \sigma^2\mu^{n-1} \frac{1-\mu^n}{1-\mu} = \lambda \lambda^{n-1} \frac{1-\lambda^n}{1-\lambda} = \lambda^n \frac{1-\lambda^n}{1-\lambda}$, if $\lambda \neq 1$; and $Var(X_n) = \sigma^2 n = \lambda n$, if $\lambda = 1$.

(c) Let $\pi_0 < 1$ denote the extinction probability. It is the smallest positive solution of the equation

$$\pi_0 = \sum_{n=0}^{\infty} \pi_0^n \frac{\lambda^n}{n!} e^{-\lambda} = e^{-\lambda} \sum_{n=0}^{\infty} \frac{(\pi_0 \lambda)^n}{n!} = e^{-\lambda} e^{\pi_0 \lambda} = e^{\lambda(\pi_0 - 1)}.$$

Below is the code and the graph of the numeric solution π_0 of this equation as a function of $\lambda > 1$.

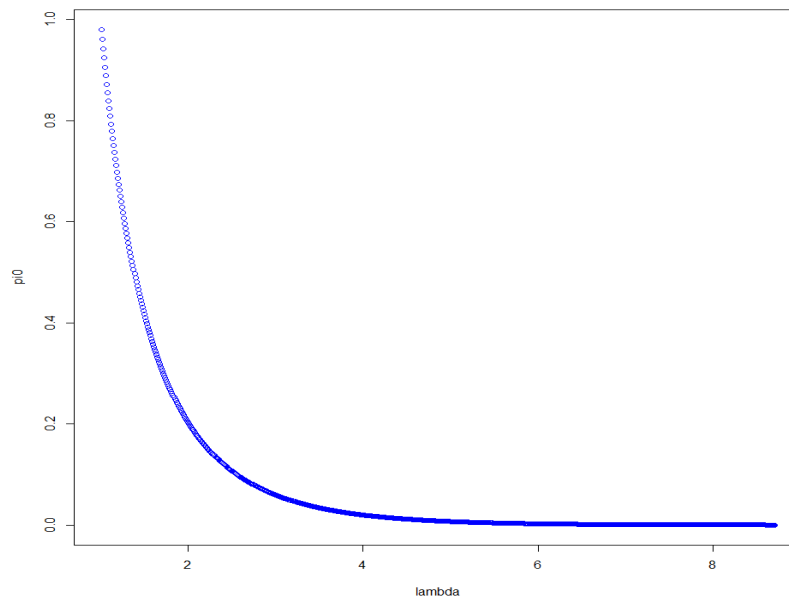
```
library(rootSolve)

lambda<- seq(1.01,8.7,0.01)
pi0<- 1:770

for (i in 1:770) {
equation<- function(x)
  x-exp(lambda[i]*(x-1))

pi0[i]<- uniroot.all(equation, c(0,0.99))
}

plot(lambda,pi0, col="blue")
```



```

> pi0
[1] 0.9801000000 0.9608919215
[3] 0.9422288852 0.9240007044
[5] 0.9061766745 0.8891012444
[7] 0.8721393433 0.8557623756
[9] 0.8396258028 0.8239578325
[11] 0.8086784805 0.7937183106
[13] 0.7791773528 0.7649519043
[15] 0.7510401355 0.7375289728
[17] 0.7242501181 0.7113162379
[19] 0.6987154931 0.6863624122
[21] 0.6742636161 0.6624665795
[23] 0.6509652371 0.6397054345
[25] 0.6286859791 0.6179043043
[27] 0.6073567259 0.5970386674
[745] 0.0003700480 0.0003700480
[747] 0.0003700480 0.0003700480
[749] 0.0003700480 0.0003700480
[751] 0.0003700480 0.0003700480
[753] 0.0003700480 0.0003700480
[755] 0.0003700480 0.0003700480
[757] 0.0003700480 0.0003700480
[759] 0.0003700480 0.0000000000
[761] 0.0000000000 0.0000000000
[763] 0.0000000000 0.0000000000
[765] 0.0000000000 0.0000000000
[767] 0.0000000000 0.0000000000
[769] 0.0000000000 0.0000000000

```

EXERCISE 8.3. (a) Let Z denote the size of male offspring. It is given that $P(Z = 0) = 0.4828$, and $P(Z = n) = (0.228292)(0.5586)^{n-1}$, $n = 1, 2, \dots$. The expected value and variance of Z are $\mu = E(Z) = (0)(0.4828) + (0.228292) \sum_{n=1}^{\infty} n(0.5586)^{n-1} = \frac{0.228292}{(1-0.5586)^2} = 1.171726$, and $\sigma^2 = \text{Var}(Z) = (0)^2(0.4828) + (0.228292) \sum_{n=1}^{\infty} n^2(0.5586)^{n-1} - (1.171726)^2 = (0.228292) \left[\frac{0.5586}{(1-0.5586)^2} + \frac{1}{(1-0.5586)^2} \right] - (1.171726)^2 = 0.45331$.

(b) The expected size and variance of the n th generation are $E(X_n) = \mu^n = (1.171726)^n$, and $\text{Var}(X_n) = \sigma^2 \mu^{n-1} \frac{1-\mu^n}{1-\mu} = (0.45331)(1.171726)^{n-1} \frac{1-(1.171726)^n}{1-1.171726} = (2.639728)(1.171726)^{n-1}((1.171726)^n - 1)$.

(c) Let π_0 denote the probability of extinction. It is found as the smallest positive solution of the equation

$\pi_0 = \pi_0^0(0.4828) + \sum_{n=1}^{\infty} \pi_0^n (0.228292)(0.5586)^{n-1} = 0.4828 + \frac{0.228292}{0.5586} \sum_{n=1}^{\infty} (0.5586\pi_0)^n =$
 $0.4828 + \frac{0.228292}{0.5586} \left(\frac{1}{1-0.5586\pi_0} - 1 \right) = 0.4828 + \frac{0.228292\pi_0}{1-0.5586\pi_0}$, which is a quadratic equation
 $0.5586\pi_0^2 - 1.0414\pi_0 + 0.4828 = 0$. The solution is $\pi_0 = 0.864304$.

EXERCISE 8.4.

(a) The mean winnings of a stake of \$1 is $\mu = (\$1)(0.3) + (\$15)(0.2) + (\$20)(0.1) + (\$0)(0.4) = \$5.3 > 1$. Since $5.3 > 1$, this the process is supercritical.

(b) The mean winning of n th bet is $\mu^n = (\$5.3)^n$, and therefore, the expected winning on the fifth bet is $\mu^5 = (\$5.3)^5 = \$4,181.96$.

(c) Let π_0 be the probability that the gambler's stake eventually turns into \$0. It is the smallest positive solution of the equation $\pi_0 = 0.4 + 0.3\pi_0 + 0.2\pi_0^{15} + 0.1\pi_0^{20}$. Solved numerically, $\pi_0 = 0.5714957$. The R script follows.

```

library(rootSolve)

equation<- function(x)
  0.4-0.7*x+0.2*x^15+0.1*x^20

Print(pi0<- uniroot.all(equation, c(0,0.99)))

0.5714957
  
```

EXERCISE 8.5. (a) The mean number of computers that are infected in one day is $\mu = \frac{1}{4}(0 + 1 + 2 + 3) = 1.5$. Since the mean is larger than one, it is a supercritical process.

(b) The average number and standard deviation of infected computers on day 10 are

$$E(X_{10}) = \mu^{10} = (1.5)^{10} = 57.66504, \text{ and}$$

$$\sqrt{\text{Var}(X_{10})} = \sqrt{\left(\frac{1}{4}(0^2 + 1^2 + 2^2 + 3^2) - (1.5)^2\right)(1.5)^9 \frac{1-(1.5)^{10}}{1-1.5}} = \sqrt{5,445.9862} = 73.7969.$$

(c) Since the process of virus spread is a supercritical branching process, the virus will not die out with probability one. The probability of its extinction π_0 is the smallest positive solution of the equation $\pi_0 = \frac{1}{4}(1 + \pi_0 + \pi_0^2 + \pi_0^3)$, or, equivalently,

$$1 - 3\pi_0 + \pi_0^2 + \pi_0^3 = (\pi_0 - 1)(\pi_0 - (-1 + \sqrt{2}))(\pi_0 - (-1 - \sqrt{2})) = 0.$$

Hence, $\pi_0 = -1 + \sqrt{2} = 0.4142$.

(d) Below we simulate the number of infected computers for 10 days. A total of 424 computers became infected in this simulation.

```

#specifying parameters
p<- c(0.25, 0.25, 0.25, 0.25)
  
```

```

N<- c()
N[1]<- 1

#specifying seed
set.seed(1088878)

#simulating offspring
for (i in 2:11) {
  Z<- 0
  for (j in 1:N[i-1]) {
    Z<- Z + sample(0:3, 1, prob=p)
  }

  N[i]<- Z

  if (N[i]==0) break
}

N

1  2  5  9 19 28 33 52 72 90 113

sum(N)

424

```

EXERCISE 8.6. (a) The code below generates the offspring of the first 20 generations of the branching process and calculates the total population size.

```

#specifying parameters
p<- c(0.1, 0.4, 0.5)
N<- c()
N[1]<- 1

#specifying seed
set.seed(377584410)

#simulating offspring
for (i in 2:20) {
  Z<- 0
  for (j in 1:N[i-1]) {
    Z<- Z + sample(0:2, 1, prob=p)
  }

  N[i]<- Z

  if (N[i]==0) break
}

N

1  1  2  4  6  8 11 15 21 30 39 56 79 101 129 181 251 345 468 667

```

In this simulation, the 20th generation size is 667 particles, which constitute the offspring of the 19th generation.

```
sum(N)
```

2415

There are a total of 2,415 particles in the 20 generations.

(b) The code below simulates a sample trajectory of this process and plots the branching process for 6 generations.

```
Library(tidyverse)

#specifying parameters
gen.max<- 6
p<- c(0.1, 0.4, 0.5)

#specifying seed
set.seed(332975)

#simulating trajectory
level.segment <- function(gen, y, branch.num) {

  branch<- data.frame(x=c(), y=c(), xend=c(), yend=c())
  gen.remaining<- gen.max-gen-1

  if (gen.remaining < 0) return(branch)

  if (branch.num > 0) {
    branch<- rbind(branch, data.frame(x=gen, y=y, xend=gen+1, yend=y),
      level.segment(gen=gen+1, y=y, branch.num=sample(0:2, 1, prob=p)))
  }

  if (branch.num > 1) {
    branch<- rbind(branch, data.frame(x=gen, y=y, xend=gen+1,
      yend=y+3^gen.remaining), level.segment(gen=gen+1, y=y+3^gen.remaining,
      branch.num=sample(0:2, 1, prob=p)))
  }

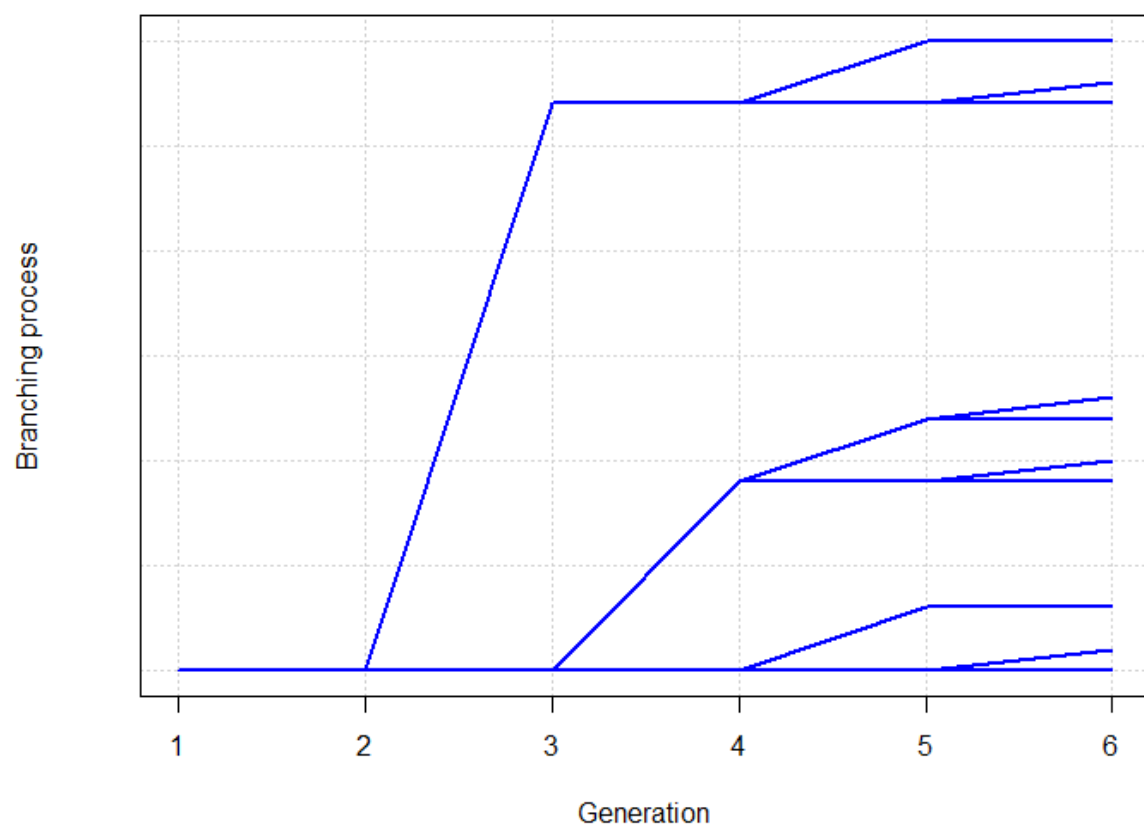
  if (branch.num > 2) {
    branch<- rbind(branch, data.frame(x=gen, y=y, xend=gen+1,
      yend=y-3^gen.remaining), level.segment(gen=gen+1, y=y-3^gen.remaining,
      branch.num=sample(0:2, 1, prob=p)))
  }

  branch
}

bp<- level.segment(1, 0, sample(0:2, 1, prob=p))

#plotting trajectory
plot(bp[,1], bp[,2], type="n", yaxt="n", xlim=c(1,6), ylim=c(range(bp)),
  xlab="Generation", ylab="Branching process", panel.first=grid())

segments(bp[,1], bp[,2], bp[,3], bp[,4], lwd=2, col="blue")
```



CHAPTER 9

EXERCISE 9.1.

$$\rho(B(s), B(t)) = \frac{\text{Cov}(B(s), B(t))}{\sqrt{\text{Var}(B(s))}\sqrt{\text{Var}(B(t))}} = \frac{\min(s, t)}{\sqrt{s}\sqrt{t}} = \begin{cases} \frac{s}{\sqrt{s}\sqrt{t}} = \sqrt{\frac{s}{t}}, & \text{if } s < t, \\ \frac{t}{\sqrt{s}\sqrt{t}} = \sqrt{\frac{t}{s}}, & \text{if } t < s \end{cases} = \sqrt{\frac{\min(s, t)}{\max(s, t)}}.$$

EXERCISE 9.2. (a) $X(t) = t B\left(\frac{1}{t}\right)$ has mean $E(X(t)) = tE\left(B\left(\frac{1}{t}\right)\right) = 0$ and variance

$\text{Var}(X(t)) = t^2 \left(\frac{1}{t}\right) = t$. Also, it has a normal distribution with independent and stationary increments since $B(t)$ does. Therefore, $X(t)$ is a standard Brownian motion.

(b) $Y(t) = \alpha B_1(t) + \sqrt{1 - \alpha^2} B_2(t)$ has a normal distribution as a linear combination of two normally distributed random variables. It also has independent and stationary increments, inherited from $B_1(t)$ and $B_2(t)$. The mean is $E(Y(t)) = \alpha E(B_1(t)) + \sqrt{1 - \alpha^2} E(B_2(t)) = 0$, and the variance is $\text{Var}(Y(t)) = \text{Var}(\alpha B_1(t) + \sqrt{1 - \alpha^2} B_2(t)) = \alpha^2 \text{Var}(B_1(t)) + (1 - \alpha^2) \text{Var}(B_2(t)) = \alpha^2 t + (1 - \alpha^2) t = t$, thus, $Y(t)$ is a standard Brownian motion.

EXERCISE 9.3.

(a) $P(0 < B(1) < 1, 1 < B(3) - B(1) < 3) = P(0 < B(1) < 1)P(1 < B(3) - B(1) < 3)$
(by independence of increments)

$= P(0 < B(1) < 1)P(1 < B(2) < 3)$ (by stationarity of increments)

$= P(0 < B(1) < 1)P(1 < \sqrt{2}B(1) < 3)$ (by rescaling of Brownian motion)

$= (\Phi(1) - \Phi(0)) \left(\Phi\left(\frac{3}{\sqrt{2}}\right) - \Phi\left(\frac{1}{\sqrt{2}}\right) \right) = 0.076053.$

(b) $P(0 < B(1) < 1, 1 < B(2) < 3) = P(0 < B(1) < 1, 1 - B(1) < B(2) - B(1) < 3 - B(1))$
 $= \int_0^1 P(1 - x < B(2) - B(1) < 3 - x \mid B(1) = x) f_{B(1)}(x) dx$

$= \int_0^1 P(1 - x < B(1) < 3 - x) f_{B(1)}(x) dx$ (by independence and stationarity of increments)

$$= \int_0^1 \int_{1-x}^{3-x} \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dy dx = 0.2198108.$$

```
func<- function(x) (pnorm (3-x,0,1)-pnorm(1-x,0,1))*pnorm(x,0,1)
```

```
integrate(func, 0, 1)
```

0.2198108 with absolute error < 2.4e-15

(c) $P(0 < B(1) < 1, 0 < B(2) < \infty) = P(0 < B(1) < 1, -B(1) < B(2) - B(1) < \infty) =$
 $\int_0^1 P(-x < B(2) - B(1) < \infty \mid B(1) = x) f_{B(1)}(x) dx = \int_0^1 P(-x < B(1) < \infty) f_{B(1)}(x) dx$
(by independence and stationarity of increments)

$= \int_0^1 P(-\infty < B(1) < x) f_{B(1)}(x) dx$ (by the symmetry of normal distribution)

$$= \int_0^1 \Phi(x) d\Phi(x) = \frac{1}{2} (\Phi^2(1) - \Phi^2(0)) = 0.228931.$$

EXERCISE 9.4. The distribution of $B(s) + B(t)$ is normal as the sum of two normally distributed random variables. The mean is $E(B(s) + B(t)) = E(B(s)) + E(B(t)) = 0$, and variance is $\text{Var}(B(s) + B(t)) = \text{Var}(2B(s) + B(t) - B(s)) = \text{Var}(2B(s)) + \text{Var}(B(t) - B(s))$ (by independence of increments) $= 4\text{Var}(B(s)) + \text{Var}(B(t - s))$ (by stationarity of increments) $= 4s + t - s = 3s + t$. Alternatively, $\text{Var}(B(s) + B(t)) = \text{Var}(B(s)) + 2\text{cov}(B(s), B(t)) + \text{Var}(B(t)) = s + 2\min(s, t) + t = 3s + t$.

EXERCISE 9.5. (a) For $x \geq 0$, $P(|B(t)| \leq x) = P(-x \leq B(t) \leq x) = \int_{-x}^x \frac{1}{\sqrt{2\pi t}} e^{-\frac{u^2}{2t}} du$
 $= 2 \int_0^{x/\sqrt{t}} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz = 2 \left(\Phi\left(\frac{x}{\sqrt{t}}\right) - \Phi(0) \right) = 2 \left(\Phi\left(\frac{x}{\sqrt{t}}\right) - \frac{1}{2} \right) = 2\Phi\left(\frac{x}{\sqrt{t}}\right) - 1 = F_{M(t)}(x).$
 (b) The density of $M(t)$ is $F'_{M(t)}(x) = 2\Phi'\left(\frac{x}{\sqrt{t}}\right) = 2 \frac{1}{\sqrt{2\pi}} e^{-\frac{(x/\sqrt{t})^2}{2}} = \sqrt{\frac{2}{\pi}} e^{-\frac{x^2}{2t}}, x \geq 0$. Therefore, the mean is computed as

$$E(M(t)) = \int_0^\infty \sqrt{\frac{2}{\pi}} x e^{-\frac{x^2}{2t}} dx = \{z = x/\sqrt{t}\} = \sqrt{\frac{2t}{\pi}} \int_0^\infty e^{-\frac{z^2}{2}} d\frac{z^2}{2} = \sqrt{\frac{2t}{\pi}}.$$

(c) The theoretical mean is $E(M(5)) = \sqrt{\frac{(2)(5)}{\pi}} = 1.784124$. Below we simulate 1,000 trajectories of a standard Brownian motion on the interval $[0, 5]$ and calculate the sample mean of the maximum $\hat{E}(M(t))$.

```
#defining Brownian motion as matrix
BM<- matrix(NA, nrow=5000, ncol=1000)

#specifying seed
set.seed(8022022)

#simulating trajectories
for (j in 1:1000) {
  BM[,j]<- 0

  for (i in 2:5000)
    BM[i,j]<- BM[i-1,j] + sqrt(0.001)*rnorm(1)
}

#computing maximum of each trajectory
max.BM<- c()
for (j in 1:1000)
  max.BM[j]<- max(BM[,j])

#computing mean of maxima
mean(max.BM)

1.780714
```

EXERCISE 9.6. (a) Consider the process $\{-B(t), t \geq 0\}$. It has a normal distribution with independent and stationary increments since $B(t)$ does. The mean is $E(-B(t)) = -E(B(t)) = 0$, and variance is $Var(-B(t)) = (-1)^2 Var(B(t)) = Var(B(t)) = t$. Therefore, it is a standard Brownian motion.

(b) To find the cumulative distribution function of $\min_{0 \leq s \leq t} B(s)$, we write for $x \leq 0$,

$$\begin{aligned} P\left(\min_{0 \leq s \leq t} B(s) \leq x\right) &= 1 - P\left(\min_{0 \leq s \leq t} B(s) > x\right) = 1 - P(B(s) > x, \forall s \in [0, t]) \\ &= 1 - P(-B(s) < -x, \forall s \in [0, t]) = 1 - P\left(\max_{0 \leq s \leq t} (-B(s)) < -x\right) \\ &= 1 - \left(2\Phi\left(\frac{-x}{\sqrt{t}}\right) - 1\right) = 2\left(1 - \Phi\left(\frac{-x}{\sqrt{t}}\right)\right) = 2\Phi\left(\frac{x}{\sqrt{t}}\right) \text{ (by symmetry).} \end{aligned}$$

(c) $P\left(\min_{0 \leq s \leq 5} B(s) < -3\right) = 2\Phi\left(\frac{-3}{\sqrt{5}}\right) = 0.179712.$

(d) The code below generates 1,000 trajectories of a standard Brownian motion and calculates the empirical probability that the minimum is less than -3 on the interval [0,5].

```
#specifying seed
set.seed(2541165)

#simulating trajectories

BM<- matrix(NA, nrow=500, ncol=1000)

for (j in 1:1000) {
  BM[1,j]<- 0

  for (i in 2:500)
    BM[i,j]<- BM[i-1,j] + sqrt(0.01)*rnorm(1)
}

#computing indicator of minimum < -3
ind<- c()

for(j in 1:1000)

  ind[j]<- ifelse(min(BM[,j])< -3, 1,0)

sum(ind)/1000

0.179
```

EXERCISE 9.7. (a) Consider the process

$$X(t) = \begin{cases} (1-t)B\left(\frac{t}{1-t}\right), & \text{if } 0 \leq t < 1, \\ 0, & \text{if } t = 1, \end{cases}$$

where $\{B(t), t \geq 0\}$ is a standard Brownian motion. Note that $X(0) = B(0) = 0 = X(1)$, so the process is tied at the endpoints of a unit interval. It has a normal distribution since $B(\cdot)$ is normally distributed. Its mean is $E(X(t)) = (1-t)E\left(B\left(\frac{t}{1-t}\right)\right) = 0$, and variance is $Var\left((1-t)B\left(\frac{t}{1-t}\right)\right) = (1-t)^2\left(\frac{t}{1-t}\right) = t(1-t)$. The covariance is

$$\begin{aligned} Cov(X(s), X(t)) &= (1-s)(1-t) Cov\left[B\left(\frac{s}{1-s}\right), B\left(\frac{t}{1-t}\right)\right] = (1-s)(1-t) \min\left(\frac{s}{1-s}, \frac{t}{1-t}\right) \\ &= \begin{cases} (1-s)(1-t) \frac{s}{1-s}, = s-st, & \text{if } s < t, \\ (1-s)(1-t) \frac{t}{1-t}, = t-st, & \text{if } t < s \end{cases} = \min(s, t) - st. \end{aligned}$$

Thus, $X(t)$ is a Brownian bridge.

(b) Consider the process $\{B(t) = (1+t)X\left(\frac{t}{1+t}\right), t \geq 0\}$, where $\{X(t), 0 \leq t \leq 1\}$ is a Brownian bridge. It is normally distributed since $X(t)$ is. The mean is $E(B(t)) = (1+t)E\left(X\left(\frac{t}{1+t}\right)\right) = 0$, and the variance is $Var(B(t)) = (1+t)^2 Var\left(X\left(\frac{t}{1+t}\right)\right) = (1+t)^2 \left(\frac{t}{1+t}\right) \left(1 - \frac{t}{1+t}\right) = t$. The covariance is $Cov(B(s), B(t)) = (1+s)(1+t) Cov\left(X\left(\frac{s}{1+s}\right), X\left(\frac{t}{1+t}\right)\right) = (1+s)(1+t) \left[\min\left(\frac{s}{1+s}, \frac{t}{1+t}\right) - \left(\frac{s}{1+s}\right) \left(\frac{t}{1+t}\right) \right] = (1+s)(1+t) \left[\frac{\min(s, t)}{1 + \min(s, t)} - \left(\frac{s}{1+s}\right) \left(\frac{t}{1+t}\right) \right] = \frac{(1+s)(1+t) \min(s, t)}{1 + \min(s, t)} - st = (1 + \max(s, t)) \min(s, t) - st = \min(s, t) + st - st = \min(s, t)$. The above properties indicate that $\{B(t), t \geq 0\}$ is a standard Brownian motion.

(c) Let $\{B(t) = X(t) + tZ, 0 \leq t \leq 1\}$ where $\{X(t), 0 \leq t \leq 1\}$ is a Brownian bridge and Z is an independent standard normal random variable. The distribution of $B(t)$ is normal being a linear combination of two normally distributed random variables. The mean is $E(B(t)) = E(X) + tE(Z) = 0$, and the variance is $Var(B(t)) = Var(X(t) + tZ) = Var(X(t)) + t^2 Var(Z) = t(1-t) + t^2 = t$. The covariance is $Cov(B(s), B(t)) = Cov(X(s) + sZ, X(t) + tZ) = Cov(X(s), X(t)) + st Var(Z) = \min(s, t) - st + st = \min(s, t)$. These indicate that the process $\{B(t), 0 \leq t \leq 1\}$ is a standard Brownian motion.

EXERCISE 9.8. (a) For $0 \leq s < t$, $P(B(s) \leq x \mid B(t) = w) = \int_{-\infty}^x f_{B(s) \mid B(t)}(u \mid w) du$

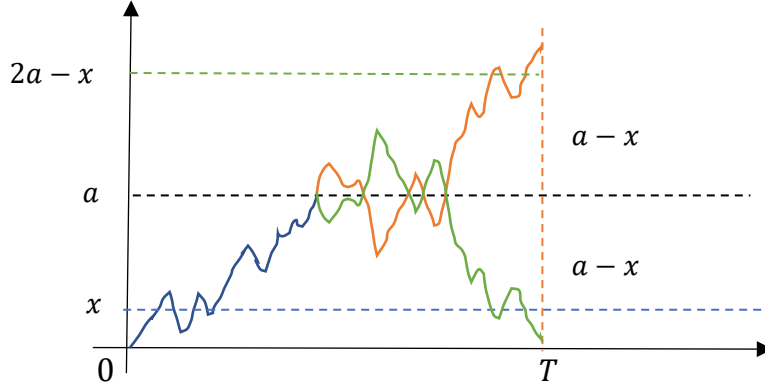
$$= \int_{-\infty}^x \frac{f_{B(s), B(t)-B(s)}(u, w-u)}{f_{B(t)}(w)} du = \int_{-\infty}^x \frac{f_{B(s)}(u) f_{B(t)-B(s)}(w-u)}{f_{B(t)}(w)} du \quad (\text{by independence of increments})$$

$$= \int_{-\infty}^x \frac{\frac{1}{\sqrt{2\pi s}} e^{-\frac{u^2}{2s}} \frac{1}{\sqrt{2\pi(t-s)}} e^{-\frac{(w-u)^2}{2(t-s)}}}{\frac{1}{\sqrt{2\pi t}} e^{-\frac{w^2}{2t}}} du \quad (\text{by stationarity of increments})$$

$$= \int_{-\infty}^x \frac{1}{\sqrt{2\pi}} \sqrt{\frac{t}{s(t-s)}} e^{-\frac{(u-\frac{sw}{t})^2}{2s(t-s)/t}} du, \text{ which means that the conditional distribution is normal with mean } \frac{s}{t} B(t) \text{ and variance } \frac{s}{t} (t-s).$$

(b) Brownian motion, conditional on the value of the endpoint $B(t) = 0$ is indeed a Brownian bridge on the interval $[0, t]$. It is normally distributed with independent and stationary increments (inherited from the Brownian motion). Its mean is $\frac{s}{t}B(t) = 0$ and variance is $\frac{s}{t}(t - s) = s\left(1 - \frac{s}{t}\right)$.

EXERCISE 9.9. (a) As seen from the picture below, $P(M(T) \geq a, B(T) \leq x) = P(M(T) \geq a, B(T) \geq 2a - x)$. That is, by the reflection principle, once the Brownian motion hits level a , the probability to end up at time T at x or below is the same as the probability of ending up at or above $2a - x$.



And if the Brownian motion ends up at or above $2a - x$ at time T , its maximum is definitely above a , so the event $M(T) \geq a$ can be omitted. Hence, $P(M(T) \geq a, B(T) \leq x) = P(B(T) \geq 2a - x)$, where $a > 0$ and $0 \leq x \leq a$.

(b) The joint cdf of $M(T)$ and $B(T)$ can be obtained as follows:

$$P(M(T) \leq a, B(T) \leq x) = P(B(T) \leq x) - P(M(T) \geq a, B(T) \leq x) = P(B(T) \leq x) - P(B(T) \geq 2a - x) = \Phi\left(\frac{x}{\sqrt{T}}\right) - \left(1 - \Phi\left(\frac{2a-x}{\sqrt{T}}\right)\right), \quad a > 0 \text{ and } 0 \leq x \leq a.$$

The joint density of $M(T)$ and $B(T)$ is found by differentiating the joint cdf with respect to a and x :

$$\begin{aligned} f_{M(T), B(T)}(a, x) &= \frac{\partial^2}{\partial a \partial x} \left[\Phi\left(\frac{x}{\sqrt{T}}\right) - \left(1 - \Phi\left(\frac{2a-x}{\sqrt{T}}\right)\right) \right] = \frac{2}{\sqrt{T}} \frac{1}{\sqrt{2\pi}} \frac{2(2a-x)}{2T} e^{-\frac{(2a-x)^2}{2T}} \\ &= \frac{2(2a-x)}{T\sqrt{2\pi T}} e^{-\frac{(2a-x)^2}{2T}}, \quad a > 0, 0 \leq x \leq a. \end{aligned}$$

The conditional density of $M(T)$ given that $B(T) = x$ is calculated as follows:

$$f_{M(T)|B(T)}(a|x) = \frac{f_{M(T), B(T)}(a, x)}{f_{B(T)}(x)} = \frac{\frac{2(2a-x)}{T\sqrt{2\pi T}} e^{-\frac{(2a-x)^2}{2T}}}{\frac{1}{\sqrt{2\pi T}} e^{-\frac{x^2}{2T}}} = \frac{2(2a-x)}{T} e^{-\frac{4a^2-4ax}{2T}}$$

$$= \frac{2(2a - x)}{T} e^{-\frac{2a(a-x)}{T}}, \quad a > 0, 0 \leq x \leq a.$$

(c) Letting $x = 0$ in the above formula for conditional density, we obtain the density of $M_{BB}(T)$, the maximum of a Brownian bridge on the interval $[0, T]$.

$$f_{M_{BB}(T)}(a) = \frac{4a}{T} e^{-\frac{2a^2}{T}}, \quad a > 0.$$

(d) The expected value of $M_{BB}(T)$ is derived as

$$\begin{aligned} E(M_{BB}(T)) &= \int_0^\infty a f_{M_{BB}(T)}(a) da = \int_0^\infty a \frac{4a}{T} e^{-\frac{2a^2}{T}} da = \left\{ z = \frac{2a}{\sqrt{T}} \right\} \\ &= \frac{\sqrt{T}}{2} \int_0^\infty z^2 e^{-\frac{z^2}{2}} dz = \frac{\sqrt{T}}{2} \cdot \frac{1}{2} \sqrt{2\pi} \int_{-\infty}^\infty \frac{z^2}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz = \frac{1}{2} \sqrt{\frac{\pi T}{2}}. \end{aligned}$$

The integral is equal to one since it is the expression for the variance of a standard normal random variable.

EXERCISE 9.10. (a) The code below plots a simulated trajectory of a two-dimensional Brownian bridge, both coordinates of which are independent Brownian bridges on the time interval $[0, T]$ where $T = (24)(60) = 1440$ minutes.

```
#defining processes as vectors
BMX<- c()
BMY<- c()
BBX<- c()
BBY<- c()

#specifying seed
set.seed(6151009)

#simulating two trajectories of Brownian motion

BMX[1]<- 0
BMY[1]<- 0

for (i in 2:1440) {
  BMX[i]<- BMX[i-1] + rnorm(1)
  BMY[i]<- BMY[i-1] + rnorm(1)
}

#computing two trajectories of Brownian bridge

for (i in 1:1440) {
  BBX[i]<- BMX[i]-i/1440*BMX[1440]
  BBY[i]<- BMY[i]-i/1440*BMY[1440]
}

#plotting two-dimensional Brownian bridge
```

```

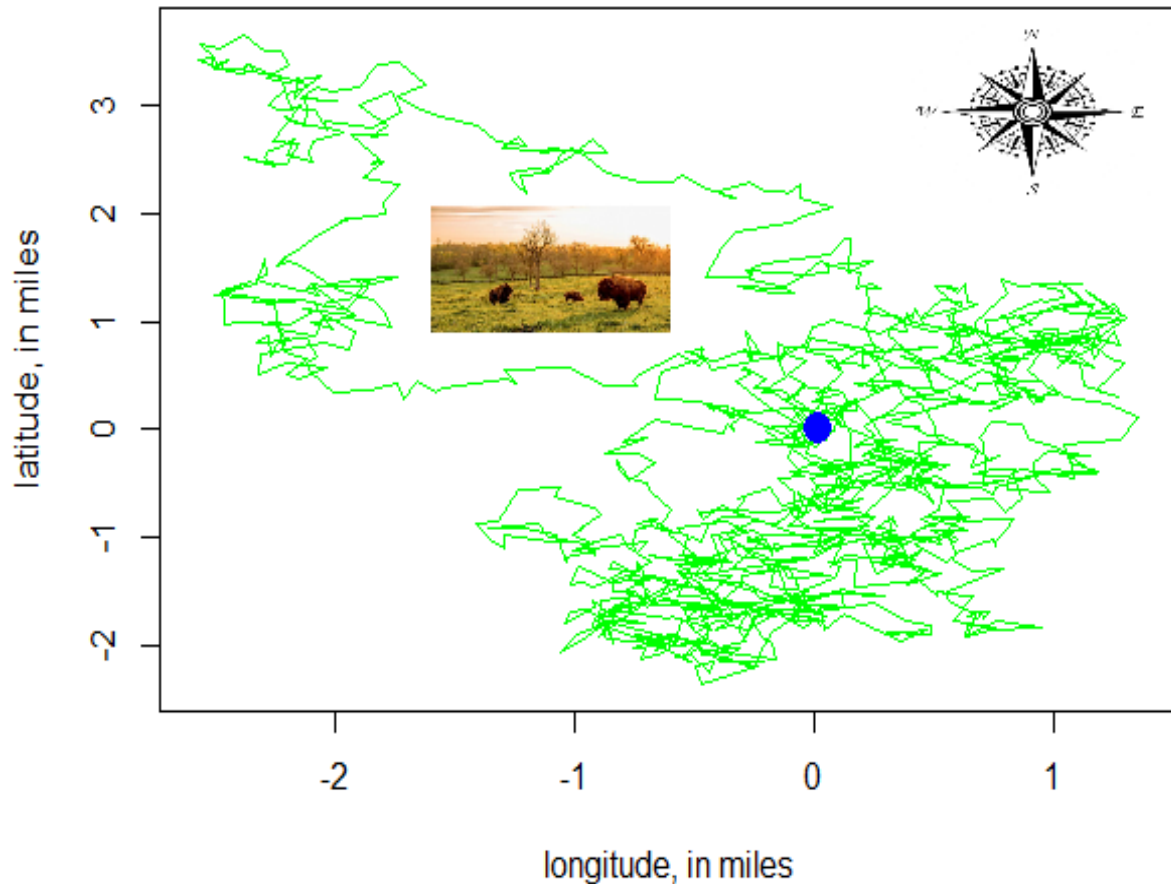
plot(BBX/10, BBY/10, type="l", col="green", xlab="longitude, in miles",
ylab="latitude, in miles")

#plotting water source in blue
points(BBX[1], BBY[1], col="blue", pch=16, cex=2)

install.packages("magick")
library(magick)
compass<- image_read("./compass.png")
bison<- image_read("./bison.png")

rasterImage(compass, 0.4, 2.1, 1.4, 3.8)
rasterImage(bison, -1.6, 0.9, -0.6, 2.1)

```



(b) From Application 9.1, we know that the expected diameter of a one-dimensional home range is $\sqrt{\frac{\pi T}{2}}$ tenths of a mile. Thus, the expected area of a two-dimensional home range is $\sqrt{\frac{\pi T}{2}} \cdot \sqrt{\frac{\pi T}{2}} = \frac{\pi T}{2} = \frac{\pi(1440)}{2} = 2261.945$ squared tenths of a mile or 22.62 squared miles.

(c) The following code simulates 1,000 trajectories and computes the sample value of the area.

```

#defining trajectories as matrices
BMX<- matrix(NA, nrow=1440, ncol=1000)
BMY<- matrix(NA, nrow=1440, ncol=1000)
BBX<- matrix(NA, nrow=1440, ncol=1000)

```

```

BBY<- matrix(NA, nrow=1440, ncol=1000)

#specifying seed
set.seed(822815)

#simulating trajectories of Brownian motion

for (j in 1:1000) {
  BMX[1,j]<- 0
  BMY[1,j]<- 0

  for (i in 2:1440) {
    BMX[i,j]<- BMX[i-1,j] + rnorm(1)
    BMY[i,j]<- BMY[i-1,j] + rnorm(1)
  }
}

#computing trajectories of Brownian bridge

for (j in 1:1000) {
  for (i in 1:1440) {
    BBX[i,j]<- BMX[i,j]-i/1440*BMX[1440,j]
    BBY[i,j]<- BMY[i,j]-i/1440*BMY[1440,j]
  }
}

#computing sample ranges
xrange<- c()
yrange<- c()

for (j in 1:1000) {
  xrange[j]<- max(BBX[,j])-min(BBX[,j])
  yrange[j]<- max(BBY[,j])-min(BBY[,j])
}

print(mean.area<- mean(xrange*yrange)/100)

```

21.91089

For this simulation, the sample value of the area is 21.91 squared miles.

EXERCISE 9.11. The following code estimates parameters and simulates a Brownian motion with drift and volatility and plots actual and simulated bird population size against time (in months).

(a) We plot the data first.

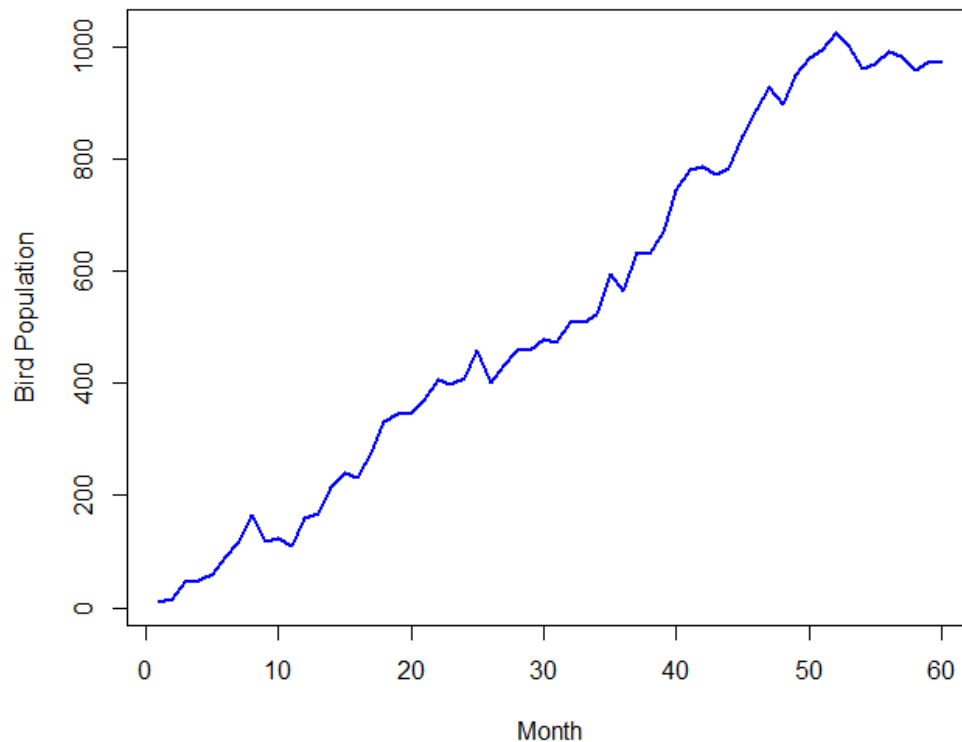
```

birds.data<- read.csv(file="./BirdPopulation.csv", header=TRUE, sep=",")

month<- birds.data$month
popl<- birds.data$population

plot(month, popl, type="l", lwd=2, cex=0, col="blue", xlab="Month",
ylab="Bird Population")

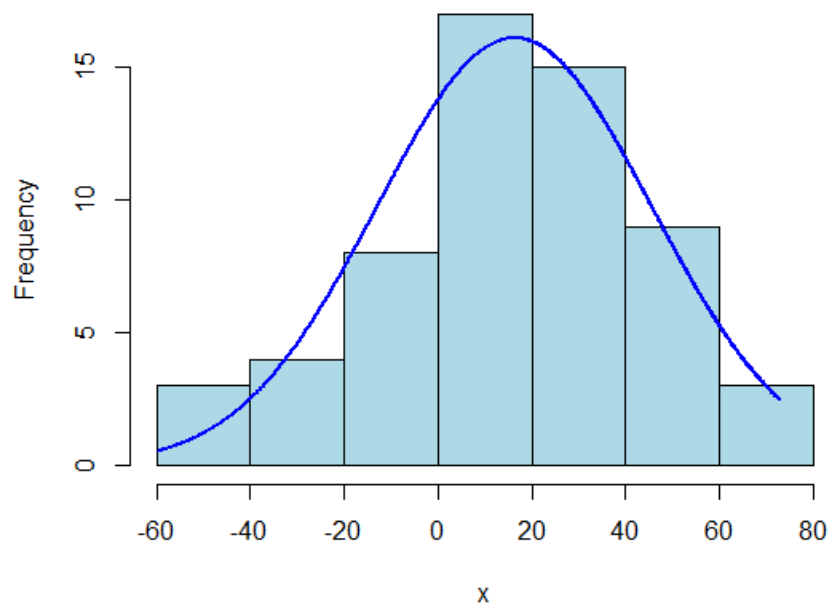
```



(b) We calculate the increments and present the histogram. The data do look normally distributed as evidenced by the bell-shaped histogram.

```
#computing increments of population size
delta.popl<- popl-c(0, head(popl, -1))
delta.popl<- delta.popl[-1]

library(rcompanion)
plotNormalHistogram(delta.popl, col="light blue")
```



(c) We estimated the drift and volatility by the method of moments estimators (which are the same as the maximum-likelihood estimators).


```
#estimating parameters
print(mu.hat<- mean(delta.popl))
```

16.33898

```
print(sigma.hat<- sd(delta.popl))
```

29.28611

(d) We simulate Brownian motion with drift and volatility and plot the actual and simulated trajectories on the same graph.

```
#simulating Brownian motion with drift and volatility
```

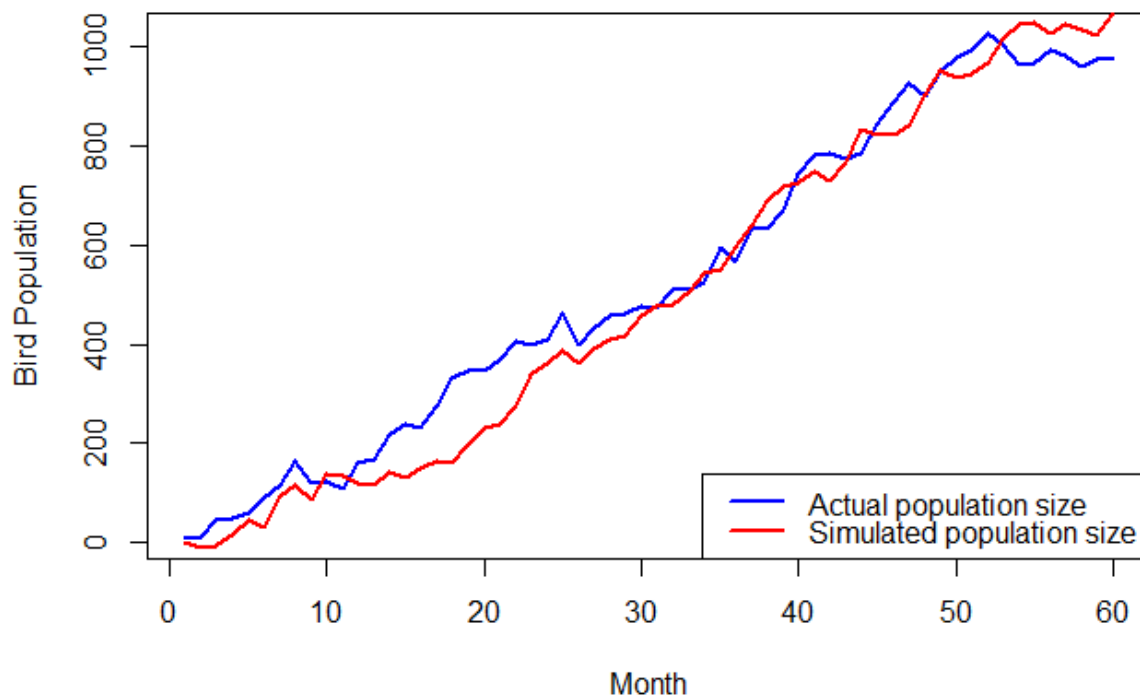
```
#specifying Brownian motion as vector
BM<- c()
```

```
#specifying initial value
BM[1]<- 0
```

```
#specifying seed
set.seed(2217626)
```

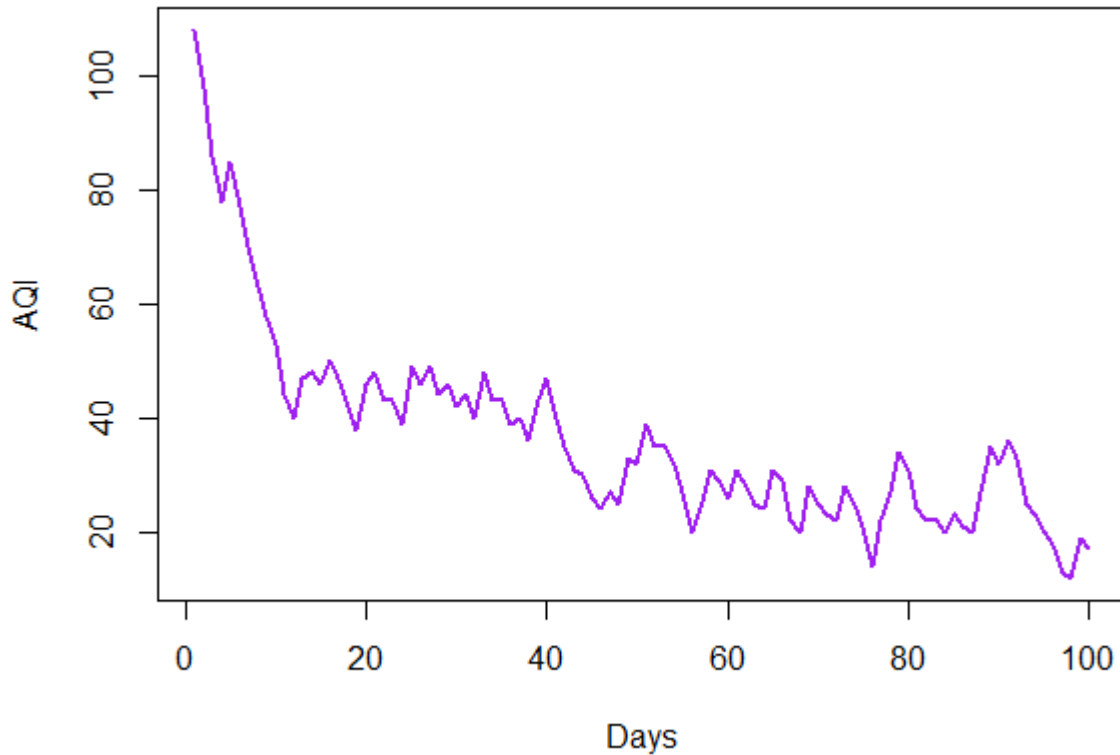
```
#simulating trajectory
for (i in 2:60)
  BM[i]<- mu.hat + BM[i-1] + sigma.hat*rnorm(1)
```

```
#plotting actual and simulated trajectories
plot(month, popl, type="l", lwd=2, cex=0, col="blue", xlab="Month",
ylab="Bird Population")
lines(month, BM, lwd=2, col="red")
legend("bottomright", c("Actual population size", "Simulated population size"),
lty=1, lwd=2, col=c("blue", "red"))
```



EXERCISE 9.12. (a) Below is the script and the plot of the data.

```
AQI.data<- read.csv(file="./AQI.csv", header=TRUE, sep=",")  
  
day<- AQI.data$day  
AQI<- AQI.data$AQI  
  
plot(day, AQI, type="l", lwd=2, cex=0, col="purple", xlab="Days", ylab="AQI")
```



We see from the graph that the values of AQI start high and then quickly decrease, suggesting that possibly a geometric Brownian motion might have a good fit.

(b) The script lines below estimate the drift and volatility coefficients of the geometric Brownian motion model.

```
#computing increments of log-AQI  
log.inc<- c()  
  
AQI1<- AQI[-1]  
AQI1.lag<- head(AQI, -1)  
log.inc<- log(AQI1/AQI1.lag)  
  
#estimating parameters  
print(mu.hat<- mean(log.inc))  
  
-0.01867594  
  
print(sigma.hat<- sd(log.inc))  
  
0.1617922
```

(c) Below we plot the actual and simulated values.

```
#simulating geometric Brownian motion

#specifying geometric Brownian motion as vector
GBM<- c()

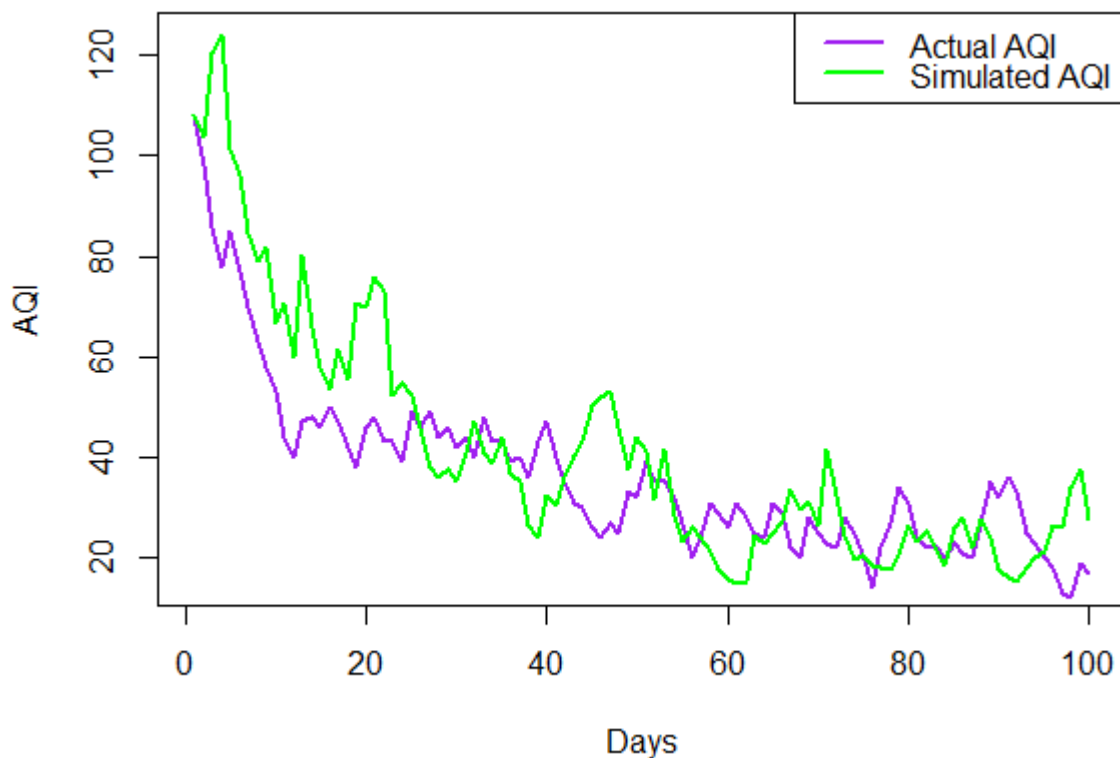
#specifying initial value
GBM[1]<- AQI[1]

#specifying seed
set.seed(34597)

#simulating trajectory
for (i in 2:100)
  GBM[i]<-GBM[i-1]*exp(mu.hat+sigma.hat*rnorm(1))

#plotting actual and simulated trajectories
plot(day, AQI, type="l", lwd=2, cex=0, col="purple", ylim=range(GBM), xlab="Days",
ylab="AQI")

lines(day, GBM, lwd=2, col="green")
legend("topright", c("Actual AQI", "Simulated AQI"), lty=1, lwd=2, col=c("purple",
"green"))
```



EXERCISE 9.13. We are given that $\mu = -0.4$, $\sigma^2 = 0.76$, $X(0) = 150$, $K = 120$, and $t = 7$.

We compute $r = \mu + \frac{\sigma^2}{2} = -0.4 + \frac{0.76}{2} = -0.02$, $A = \frac{\mu t - \ln\left(\frac{K}{X(0)}\right)}{\sigma\sqrt{t}} = \frac{(-0.4)(7) - \ln\left(\frac{120}{150}\right)}{\sqrt{(0.76)(7)}} = -1.11721$ and

$C = X(0)\Phi(A + \sigma\sqrt{t}) - e^{-rt}K\Phi(A) = (150)\Phi\left(-1.11721 + \sqrt{(0.76)(7)}\right) - e^{-(-0.02)(7)}(120)\Phi(-1.11721) = \114.21 .

EXERCISE 9.14. (a) Foreign currency exchange rates can be modeled well with an Ornstein-Uhlenbeck (OU) process because if the rates are very high, demand decreases, or, similarly, if the rates are very low, then demand increases. In both cases, the rates will revert to some long-term mean. In addition, it is reasonable to assume that the variance of exchange rates stays within certain bounds because the variance cannot grow forever in this setting.

(b) The code below fits an OU process to euro/US\$ daily exchange rate. Plots of actual and simulated trajectories are also presented.

```
exchrates.data<- read.csv(file="./Foreign_Exchange_Rates.csv", header=TRUE,
sep=",")

#estimating parameters

inc<- exchrates.data$EURO[-1]-head(exchrates.data$EURO,-1)
fit<- glm(inc ~ head(exchrates.data$EURO,-1))

theta.hat<- -fit$coefficients[2]
mu.hat<- fit$coefficients[1]/theta.hat
sigma.hat<- sigma(fit)

#simulating trajectory of OU process

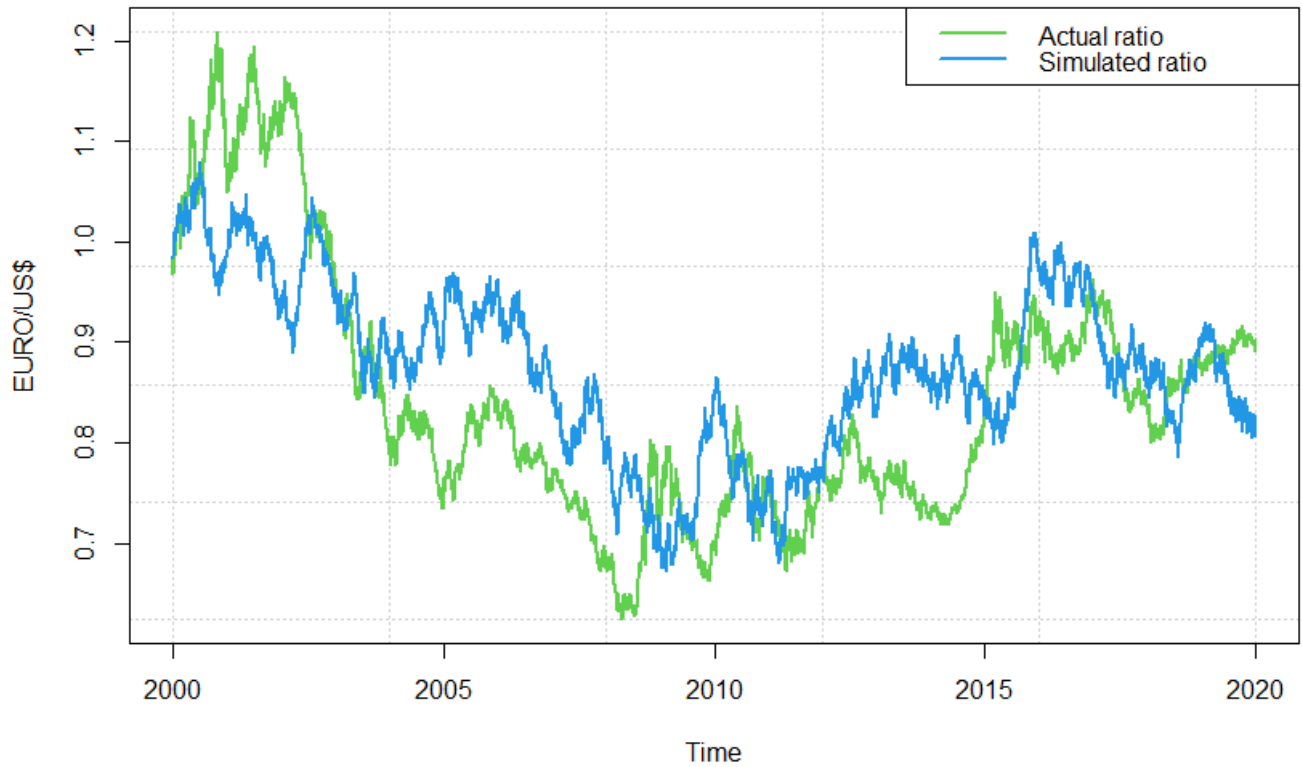
#specifying seed
set.seed(5536667)

#defining OU trajectory as vector
OU<- c()

#specifying initial value
OU[1]<- exchrates.data$EURO[1]

for (i in 2:length(exchrates.data$DATE))
  OU[i]<- OU[i-1]+theta.hat*(mu.hat-OU[i-1])+sigma.hat*rnorm(1)

#plotting trajectories
plot(as.Date(exchrates.data$DATE), exchrates.data$EURO, type="l", lty=1, lwd=2,
col=3, xlab="Time", ylab="EURO/US$", first.panel=grid())
lines(as.Date(exchrates.data$DATE), OU, lwd=2, col=4)
legend("topright", c("Actual ratio", "Simulated ratio"), lty=1, lwd=2, col=3:4)
```



EXERCISE 9.15. The process $\{B_3(t), t \geq 0\}$ has a normal distribution with independent and stationary increments since $\{B_1(t), t \geq 0\}$ and $\{B_2(t), t \geq 0\}$ have these properties. The expected value and variance of $B_3(t)$ are $E(B_3(t)) = E(\rho B_1(t) + \sqrt{1 - \rho^2} B_2(t)) = 0$, and $Var(B_3(t)) = Var(\rho B_1(t) + \sqrt{1 - \rho^2} B_2(t)) = \rho^2 Var(B_1(t)) + (1 - \rho^2) Var(B_2(t)) = \rho^2 t + (1 - \rho^2)t = t$. Thus, $\{B_3(t), t \geq 0\}$ is a standard Brownian motion.

(b) The covariance between $B_1(t)$ and $B_3(t)$ is $Cov(B_1(t), B_3(t)) = Cov(B_1(t), \rho B_1(t) + \sqrt{1 - \rho^2} B_2(t)) = \rho Var(B_1(t)) = \rho t$. The correlation coefficient is

$$\rho_{B_1(t), B_3(t)} = \frac{Cov(B_1(t), B_3(t))}{\sqrt{Var(B_1(t))} \sqrt{Var(B_3(t))}} = \frac{\rho t}{\sqrt{t} \sqrt{t}} = \rho.$$

(c) The increments $B_1(t) - B_1(s)$ and $B_2(t) - B_2(s)$ have $N(0, t - s)$ distribution, and, hence, by part (a), $B_3(t) - B_3(s)$ has the same distribution. By part (b), the increments $B_1(t) - B_1(s)$ and $B_3(t) - B_3(s)$ are correlated with the correlation coefficient ρ .

(d) We modeled the Exxon and British Petroleum (BP) historical stock prices between 4/1/2020 and 3/30/2021. The R code below plots the two processes and estimates the correlation coefficient.

```
data<- read.csv(file="./Exxon_BP_stock_prices.csv", header=TRUE, sep=",")
time<- as.Date(data$Date)
Exxon<- data$Exxon
BP<- data$BP
```

```

plot(time, Exxon, type="l", lty=1, lwd=2, col="blue", ylim=c(0,70), xlab="Time",
ylab="Stock prices", first.panel=grid())
lines(time, BP, lwd=2, col="green")
legend("bottomright", c("Exxon", "BP"), lty=1, lwd=2, col=c("blue", "green"))

```



```

Exxon.diff<-Exxon[-1]-head(Exxon,-1)
BP.diff<- BP[-1]-head(BP,-1)
cor(Exxon.diff, BP.diff)

```

0.8331595

We conclude that Exxon and BP stock prices are highly correlated with the estimated correlation coefficient of about 0.83.